64000

HP64000 Logic Development System

Model 64620S Logic State/Software Analyzer Reference Manual

HEWLETT PACKARD

CERTIFICATION

Hewlett-Packard Company certifies that this product met its published specifications at the time of shipment from the factory. Hewlett-Packard further certifies that its calibration measurements are traceable to the United States National Bureau of Standards, to the extent allowed by the Bureau's calibration facility, and to the calibration facilities of other International Standards Organization members.

WARRANTY

This Hewlett-Packard system product is warranted against defects in materials and workmanship for a period of 90 days from date of installation. During the warranty period, HP will, at its options, either repair or replace products which prove to be defective.

Warranty service of this product will be performed at Buyer's facility at no charge within HP service travel areas. Outside HP service travel areas, warranty service will be performed at Buyer's facility only upon HP's prior agreement and Buyer shall pay HP's round trip travel expenses. In all other cases, products must be returned to a service facility designated by HP.

For products returned to HP for warranty service. Buyer shall prepay shipping charges to HP and HP shall pay shipping charges to return the product to Buyer. However, Buyer shall pay all shipping charges, duties, and taxes for products returned to HP from another country.

HP warrants that its software and firmware designated by HP for use with an instrument will execute its programming instructions when properly installed on that instrument. HP does not warrant that the operation of the instrument, or software, or firmware will be uninterrupted or error free.

LIMITATION OF WARRANTY

The foregoing warranty shall not apply to defects resulting from improper or inadequate maintenance by Buyer, Buyer-supplied software or interfacing, unauthorized modification or misuse, operation outside of the environmental specifications for the product, or improper site preparation or maintenance.

NO OTHER WARRANTY IS EXPRESSED OR IMPLIED. HP SPECIFICALLY DISCLAIMS THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

EXCLUSIVE REMEDIES

THE REMEDIES PROVIDED HEREIN ARE BUYER'S SOLE AND EXCLUSIVE REMEDIES. HP SHALL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, WHETHER BASED ON CONTRACT, TORT, OR ANY OTHER LEGAL THEORY.

ASSISTANCE

Product maintenance agreements and other customer assistance agreements are available for Hewlett-Packard products.

For any assistance, contact your nearest Hewlett-Packard Sales and Service Office.

CW&A 2/81

FOLD HERE

HEWLETT PACKARD

NO POSTAGE NECESSARY IF MAILED IN THE UNITED STATES

BUSINESS REPLY CARD
FIRST CLASS PERMIT NO. 1303 COLORADO SPRINGS, COLORADO

POSTAGE WILL BE PAID BY ADDRESSEE

HEWLETT-PACKARD
Logic Product Support Dept.
Attn: Technical Publications Manager
Centennial Annex - D2
P.O. Box 617
Colorado Springs, Colorado 80901-0617

FOLD HERE

Your cooperation in completing and returning this form will be greatly appreciated. Thank you.

READER COMMENT SHEET

Operating Manual, Model 64620S
Logic State/Software Analyzer Reference Manual
64620-90905, March 1984

Your comments are important to us. Please answer this questionaire and return it to us. Circle the number that best describes your answer in questions 1 through 7. Thank you.

1. The information in this book is complete:

Doesn't cover enough (what more do you need?)	1 2 3 4 5	Covers everything

2. The information in this book is accurate:

Too many errors	1 2 3 4 5	Exactly right

3. The information in this book is easy to find:

I can't find things I need	1 2 3 4 5	I can find info quickly

4. The Index and Table of Contents are useful:

Helpful	1 2 3 4 5	Missing or inadequate

5. What about the "how-to" procedures and examples:

No help	1 2 3 4 5	Very helpful
Too many now	1 2 3 4 5	I'd like more

6. What about the writing style:

Confusing	1 2 3 4 5	Clear

7. What about organization of the book:

Poor order	1 2 3 4 5	Good order

8. What about the size of the book:

too big/small	1 2 3 4 5	Right size

Comments: _____

_____

_____

_____

Particular pages with errors?

_____

Name (optional): _____
Job title: _____
Company: _____
Address: _____

Note: If mailed outside U.S.A., place card in envelope. Use address shown on other side of this card.

Logic State/Software
Analyzer Reference Manual

Printing History

Each new edition of this manual incorporates all material updated since the previous edition. Manual change sheets are issued between editions, allowing you to correct or insert information in the current edition.

The part number on the back cover changes only when each new edition is published. Minor corrections or additions may be made as the manual is reprinted between editions. Vertical bars in a page margin indicate the location of reprint corrections.

First Printing ..... June 1982 (P/N 64620-90902)
Second Edition ..... November 1983 (P/N 64620-90903)
Third Edition ..... March 1984 (P/N 64620-90905)

Table of Contents

Table of Contents (Cont'd)

Table of Contents (Cont'd)

Table of Contents (Cont'd)

Table of Contents (Cont'd)

Table of Contents (Cont'd)

List of Illustrations

List of Illustrations (Cont'd)

List of Tables

Chapter 1
GENERAL INFORMATION

INTRODUCTION

This manual describes how to operate the Hewlett-Packard Model 64620A Logic State/Software Analyzer designed for use in the HP 64000 Logic Development System. Throughout this manual, the Logic State/Software Analyzer will be called the state/software analyzer.

MANUAL CONTENTS

This first chapter outlines the content and organization of the manual and describes the state/software analyzer. Performance specifications and operating characteristics of the analyzer are listed in Tables 1-1 and 1-2 at the end of this chapter.

Chapter 2 provides instructions for installing the board assemblies of the state/software analyzer and for making a set of operators flexible discs (if required).

Chapter 3 is called Getting Started. Perform the procedures in this chapter as soon as possible. They will help you become familiar with the state/software analyzer operation.

Chapter 4 provides a programming model of the state/software analyzer and describes its software architecture. This chapter will be most helpful after you are comfortable with the analyzer and have performed several basic measurements.

Chapter 5 provides a brief discussion of keyboard operation and explains how to enter commands using the directed syntax features within various functional modes of the state/software analyzer.

Chapter 6 describes the format specification and shows how to assign labels to identify signals on the input probe lines.

Chapter 7 describes the map specification. It shows how to set up a map, how to assign symbols that represent values found on labeled inputs from the format specification, how to enter symbols and corresponding values or ranges into the map, and how to make the analyzer default to the symbols of the map whenever a particular label is used.

Chapter 8 shows how to set up trigger, store, and count specifications for tracing state flow. Individual sections in this chapter describe how to make basic trace measurements, how to use all symbols, how to make overview measurements, and how to use the sequence elements to qualify and window your measurements, how to make intermodule measurements (involving the state/software analyzer with other analysis modules), and how to use the state/software analyzer to drive control signals to the system under test and/or to peripheral test instruments.

The last chapters of this manual show you how to read the types of information displayed by the state/software analyzer, such as lists of states captured in trace memory, and overviews of system activity during software execution.

Appendix A lists and defines all of the error and status messages that can appear on the screen when you are using the state/software analyzer. Appendix B is a glossary of the names of softkeys that you will use to formulate commands when operating the state/software analyzer. Appendix C shows a list of the ASCII character set that is supported by the state/software analyzer.

STATE/SOFTWARE ANALYZER DESCRIPTION

The state/software analyzer will perform trace measurements so you can analyze state and process flow in wide-word microprogrammed state machines and overview measurements so you can analyze performance in software-intensive systems. The state/software analyzer can be installed within a 64000 system cluster station, or in a Development Station with a flexible disc drive as a stand-alone state/software analyzer. It can be modularly expanded to have from 20 to 120 channels of input information obtained through multiples of 20-bit probes.

The state/software analyzer can be configured to satisfy a variety of measurement requirements. General-purpose and dedicated interfaces are available that greatly simplify connection to target systems. Components that can be included in a state/software analyzer configuration are as follows:

a. Control Board (HP Part No. 64621A). One required for each state/software analyzer.

b. 40-channel Acquisition Board (HP Part No. 64622A). Optional. Up to three of these boards can be included in a state/software analyzer.

c. 20-channel Acquisition And Ranging Board (HP Part No. 64623A). Optional. Up to three of these boards can be included in a state/software analyzer. This board can only perform ranging and overview measurements when it receives its input from probe pod 1.

d. Input probing:

   1. General-purpose clock pod. One required for each state/software analyzer.

   2. General-purpose data pod. One required for each 20 channels of data acquisition.

   OR

   3. General-purpose preprocessor with dedicated preprocessor interface. Provides connections (and demultiplexing, if required) to the system under test.

   4. Emulation Bus Preprocessor. Provides the feature set of the Model 64620S to the emulation user. It connects directly to the emulation bus for analysis of all emulation software entirely in emulation memory.

e. Software for state/software analyzer and operating system.

TRACE MEMORY

The memory of the state/software analyzer is 256 states deep. It stores each state captured by the input probes. These states can be acquired by using comprehensive qualifiers and windowing. The trace memory also stores measurements of time or states so you can observe the periods between occurrences of interest or identify occurrences of selected states. Additionally, the trace memory stores the status of the sequence and window elements.

ENTERING YOUR COMMANDS

The syntax for command entries is directed by the state/software analyzer through the general-purpose keyboard. You can make entries using the softkey label line. It will present symbols which you have defined in your maps, and (with a memory expander) symbols defined in absolute files under development. You can also use the keyboard to enter absolute values, if desired.

THE FORMAT SPECIFICATION

The format specification is where you identify the hardware and software configuration of the system to be tested. Format parameters set up the way the state/software analyzer interprets the signals from the clock and data pods. The format specification provides two types of displays: data_pods display and data label displays (one for each label). These displays show the labels and clocks used by the state/software analyzer during data acquisition. The labels identify the types of information carried by the input bits. The labels are used when making entries and displaying readouts. The clock can be derived from a variety of activities found on up to eight OR'd channels. These channels can carry clock pulses or qualifier logic levels.

If you are using an HP dedicated preprocessor as a probe, the analyzer will automatically set up your format specification with the proper data labels and clock identifiers for the associated interface.

If you are using the general-purpose probes, the format specification will be set up with a default configuration which you can modify to correspond to the system you are testing.

THE MAP SPECIFICATION

This is the specification where you create maps of symbols. The symbols represent values to be found on labeled sets of bits. A symbol may represent a single value, a value in which some of the bits are don't care bits, or a range of values. You can define symbols to represent addresses on an address bus, instructions on a data bus, types of transactions on status lines, or any other activity desired. You can define symbols on the maps by direct keyboard entry, or by instructing the analyzer to look up link_symbols in an absolute file under development and place the associated addresses in the symbol map.

The analyzer will present symbols from your maps when you are entering specifications so you won't have to look up the absolute values, and it will use your symbols in displays of trace data captured during trace measurements.

THE TRACE SPECIFICATION

This is the specification where you enter the parameters for making your measurements. In this specification, you can set up the state/software analyzer to perform trace measurements, overview measurements, and multiple-analyzer interactive measurements. Aspects of the trace specification are described in the following paragraphs.

Trace Measurements. A trace is a measurement which collects a series of states and presents information about these states in a list or graph format. Each state describes the conditions at several points in a system under test. By collecting a series of states, you can see how conditions changed with time at each of the monitored points.

Up to eight OR'd terms (including not-terms and ranges of terms) can be used to qualify the trigger points and states to be captured in trace memory. You can also use these OR'd terms to qualify counts of states in the trace memory.

Sequence and Window Elements. The state/software analyzer has one sequence element and two window elements. Each of these elements function independently from the other two. They share up to fifteen terms to facilitate complex control functions. Each element provides an output which is either the "enable" level or the "disable" level. Each element can be set up to transition from enable to disable and transition from disable to enable at specific points in state or program flow. You can use the enable/disable levels and switching transitions to control and qualify measurement parameters in the state/software analyzer.

Overview Measurements. An overview measurement compares the results of over 4,000 separate measurements. Each measurement is classified. Up to 15 classifications (called events) can be used in a single overview measurement to represent measurement results. These events are assigned names and are stored in the overview event memory.

There are three different overview measurement modes. These modes use events to compare:

1. Values or ranges of values found on a labeled set of probes.

2. Ranges of time periods measured between start and stop points in program flow.

3. Counts or ranges of counts of state occurrences between start and stop points in program flow.

The overview measurement function can be assigned to trigger a trace of state flow if one of the overview events is found during the measurement.

Interactive Measurements. These are measurements coordinated between two or more analyzers. Softkeys in the trace specification let you set up the state/software analyzer to receive enable levels or triggers from other analyzers in the system. You can also have the state/software analyzer drive these enable labels and triggers to other analysis modules on the intermodule bus. Additionally, the state/software analyzer can provide a delay clock signal for the other analysis modules on the intermodule bus.

Generating Control Signals. The state/software analyzer can supply two output signals to the system under test: the stimulus pulse and the halt logic level. You can use the stimulus pulse to simulate interrupts at desired points in program flow. The halt line can be used to stop microprocessor operation in the system under test. The state/software analyzer can supply these signals to a preprocessor interface and/or to BNC ports on the rear panel of the mainframe.

TRACE LIST DISPLAY

The trace list displays information captured in the trace memory in a list format. The display can include lists of states captured from each of the labeled probe inputs. These lists can show values in binary, octal, decimal, and hexadecimal. The values can also be represented using the ASCII character set. Values can also be shown using symbols from your symbol maps or symbols taken from the link_symbols file of the absolute code being traced. The display can also show memory content reverse-assembled into the mnemonics of the microprocessor under test. The state/software analyzer can perform this inverse-assembly using any disassembler routine of your choice. The trace list can also show the line-by-line status of the sequence and window elements and counts of states found in program flow of the system under test.

The state/software analyzer automatically formats the trace list during initialization. The default arrangement of information in the trace list depends upon which interface is connected. If you are using the general-purpose preprocessor with a dedicated interface, the tracelist will be formatted with columns of information labeled to correspond to the type of microprocessor system under test. If you are using general-purpose probes, the display will show labels for each of the probe pods. You can arrange the information on the trace display in any desired format.

OVERVIEW DISPLAYS

The state/software analyzer presents overview displays from two memory sources: the trace memory and the overview-event memory. From the trace memory, the state/software analyzer can compose graphic displays of state values with selectable scaling. From the overview event memory, the state/software analyzer can formulate histograms, lists, and graphs. Histograms are bar graphs which show the relative occurrences of all overview events. Overview lists show the order of those occurrences. Overview graphs show patterns in the occurrences of the overview events.

Table 1-1. Specifications

Includes Models 64621A Control Board, 64622A 40-channel Acquisition Board, and 64623A 20-channel Acquisition and Ranging Board with General-Purpose Probes.

Unqualified Clock Rate: 25 MHz max.

Qualified Clock Rate (data strobe): 10 MHz max.

Time Count: Accuracy 0.1% or 40 ns, whichever is greater.

Pulse Widths, Setup and Hold Times: all polarities.

Clock Pulse Width - 20 ns min.
Clock Qualifier Setup Time - time qualifier must be present prior to active edge of clock - 20 ns max.
Clock Qualifier Hold Time - time qualifier must remain present and stable after active edge of clock - 0 ns.
Data Setup Time - time data must be present prior to active edge of clock - 30 ns max.
Data Hold Time - time data must remain present and stable after active edge of clock - 0 ns.

BNC Port Outputs (Mainframe Rear panel): programmable polarity.

Stimulus (Port 1) - TTL pulse output into 50 ohms.
Occurs at each recognized event.
Trigger Events,
Pulse Width 50 ns +/- 20 ns.
Delay from clock 225 ns +/- 25 ns.
Sequencer Events,
Pulse Width 50 ns +/- 20 ns.
Delay from clock 200 ns +/- 25 ns.
Halt (Port 2) - TTL level output into 50 ohms.
False at execute, true at event recognition or halt.
Measurement Complete - delay from clock 250 ns +/- 25 ns.
Trace Point - delay from clock 250 +/- 25 ns.

Table 1-2. Supplemental Characteristics

Memory Size:
Width - expandable to 120 channels in combinations of 20 and/or 40-channel acquisition boards (max 3 ACQ boards).
Depth - Trace Storage - 256 locations.
Overview (Model 64623A only) - 4096 locations.

Sequence: Multiple function control with windows and qualifiers, occurrence, and restart.

Clocks: 8 OR'd clocks and/or qualifiers.

Interactive Read of Trace Data: up to 4.75 MHz qualified clock rate.

Run Status:
Waiting for trigger.
Trace in process.
Overview in process.
Slow clock.
Measurement complete.

Overview Functions: (Model 64623A only).
Sequencer Windowed/Controlled.
3 Modes -
State Data.
Time Count, start to stop, 8.0 hrs max time within 40 ns or 0.1%.
Event Count, start to stop, count by one from 0 to 611,670, max count 750 X 10E+9.

3 Displays -
Overview Histogram.
Overview List.
Overview Graph.

IMB Functions (interconnection with other modules):
Master Enable (drive, receive).
Storage Enable (drive, receive).
Trigger Enable (drive, receive).
Trigger (drive, receive, drive and receive).
Delay Clock (drive).

Table 1-2. Supplemental Characteristics (Cont'd)

20-Bit Ranging (Model 64623A only):
Applicable to trace or overview functions.
Four trace ranges or up to 15 overview ranges (5 guaranteed).
Range on a contiguous subset of the 20 bits (beginning with bit 00).

Trace Count Measurement:
Windowing of time or state count.
Stored State to Stored State -
Time Count - 8.0 hrs max time within 40 ns or 0.1%.
Event Count - count by one from 0 to 611,670, max count 750 X 10E+9.

Symbol Entry and Output:
Definition of symbol maps in map specification.
Use of symbols in trace specification.
Trace list uses symbols to present symbolic display.
Each label may have its own symbol map or use symbols defined on another symbol map.

Probing Versatility:
General Purpose Probes (Models 64635A and 64636A).
General Purpose Preprocessor with dedicated interfaces.
(See Model 64650A General Purpose Preprocessor Manual.)
Emulation Bus Preprocessor module that makes the analysis features of the state analyzer module available in the emulation environment. (See Emulation Bus Preprocessor Manual.)

Chapter 2
INSTALLATION

INTRODUCTION

This chapter provides instructions for installing the state/software analysis boards in a mainframe and for connecting the probe cables to the target system in order to make measurements.

Along with the electronic hardware, you receive either a set of flexible discs or a tape cassette which contain the necessary software to operate the state/software analyzer. If you receive flexible discs, you should make user copies of them to operate the analyzer, and save the original discs as a master set for future use. The procedure for making user copies is provided in this chapter.

INSTALLATION

The state/software analyzer can include up to three different board assemblies: control board, 20-channel ranging board, and 40-channel acquisition board. These can be arranged in a variety of configurations to satisfy measurement needs. There are certain aspects of board installation which must be observed no matter which configuration you are installing. One of these is the order in which the boards must be installed. Always observe the following order of installation:

1. Install the control board in the first slot to be occupied by cards of the state/software analyzer in the card cage (slot with lowest number).

NOTE

Record the number of the slot where you installed the control board. This number will appear on the softkey label line in the measurement systems display. This number is placed on screen to identify the state/software analyzer so that you can use two state/software analyzers in the same mainframe without confusing one with the other.

2. Install the 40-channel acquisition board or boards (optional) in the next available slot(s).

NOTE

Always install all 40-channel acquisition boards in lower numbered slots than 20-channel ranging boards.

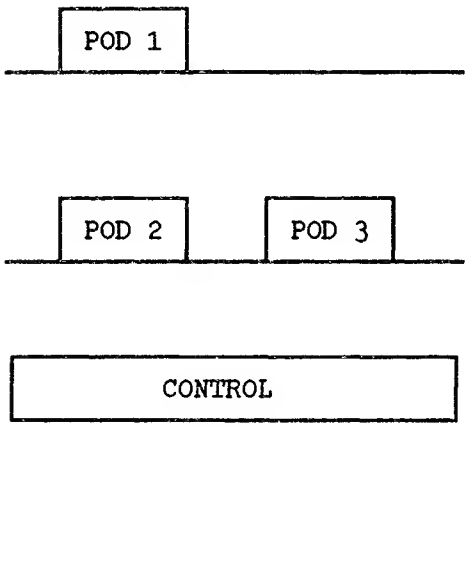
3. Install the 20-channel ranging board or boards (optional) in the next slot(s) (adjacent to the last 40-channel acquisition board).

4. Connect the probe pod cables to the board assemblies. The state/software analyzer numbers pods starting with pod 1 on the left-hand side of the board assembly installed in the highest numbered slot. Make sure pin 1 is connected to pin 1 (indicated by black wire and connector mark). Refer to the following examples of typical installations as a guide for connecting cable assemblies:

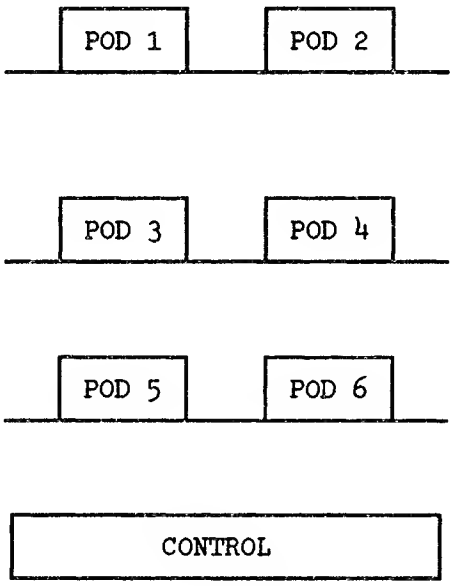
NOTE

You may need to slide the board assemblies part way out of their slots to make the cable connections.

Example 1



Example 2



5. Each probe cable has a metal ferrule for strain relief. Snap the ferrule into one of the cable clamps on the rear of the mainframe. If your mainframe does not have cable clamps, you can order them from Hewlett-Packard Co. Up to three cable clamps can be installed.

6. Connect the probe pod cables to the general-purpose probes or to the general-purpose preprocessor interface, or to any other preprocessor interface that you will use.

7. If you are using the general-purpose preprocessor interface, connect its cable to the system under test, as described in the manual for that preprocessor interface.

8. If you are using the general-purpose probes, connect the individual probe lines to the nodes to be monitored. If your system has a 20-channel ranging board installed as pod 1, you can make ranging and overview measurements on information obtained by pod 1. If your system software includes an inverse assembler for mnemonic displays, be sure that you connect the lines from your probe to the status bits of your system according to the configuration of the format specification that you will use in your measurements. All probe lines of the status bits must be contiguous.

NOTE

Refer to Chapter 6 for use of the "Activity Test" available in the Format Specification. This test will be very helpful when making connections.

9. If you are operating a system with more than one analysis module installed (timing analyzer, second state/software analyzer, etc.), connect the IMB cable from the other analysis module(s) to the state/software analyzer control board.

10. To verify correct installation, use the "Activity Test" available in the Format Specification. Refer to Chapter 6. Then run the performance verification from the flexible disc supplied. Select the (state) softkey. If configuration is illegal, an error message will appear on the display.

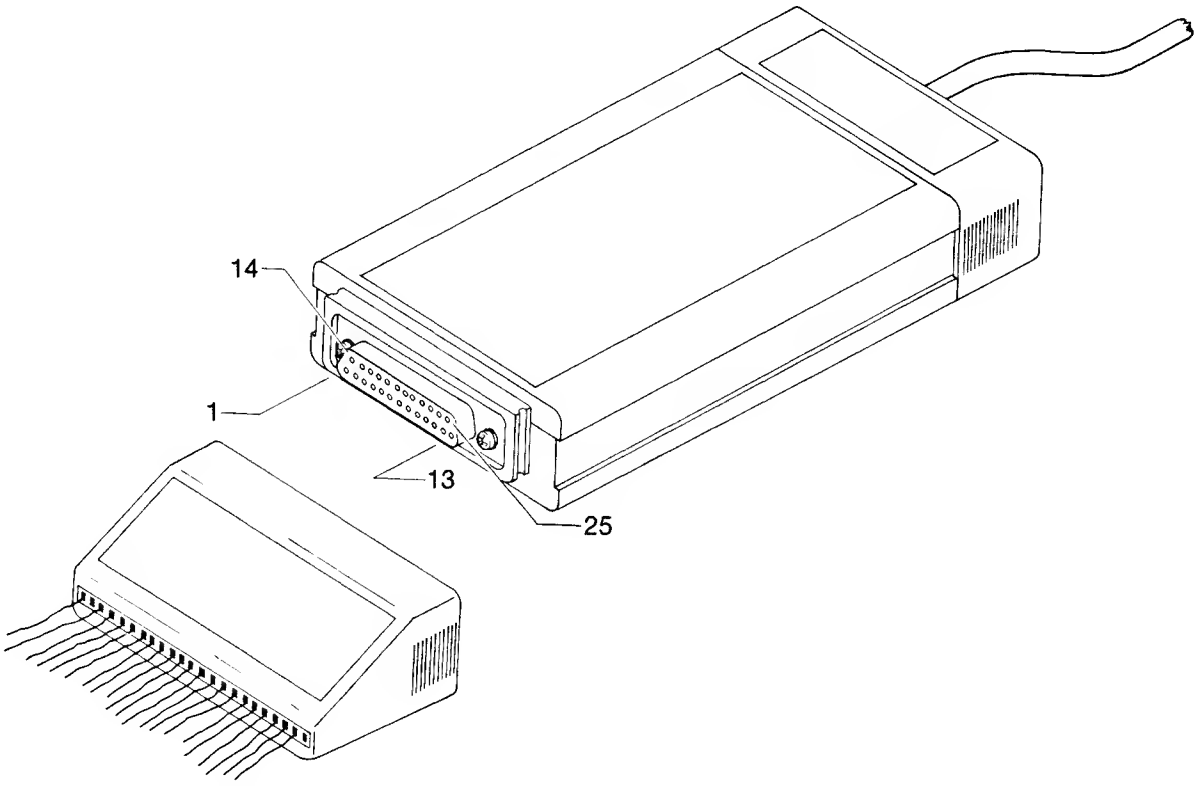
REAR-PANEL ADDRESS SWITCHES

The address switches are a set of seven switches on the mainframe rear panel. Refer to the label located beside the address switches, and set the switches as follows:

a. If you are operating this instrument as part of a system, set switches one through five to select the system address for your mainframe. Set switches six and seven to select SYSTEM BUS operation.

b. If operating as a stand-alone instrument, switches one through five are used to set the HP-IB address of the mainframe. If you are not using the HP-IB interface, set switches six and seven to one of the local mass storage (LMS) positions.

Figure 2-1 shows the signals carried on each of the pins of the general purpose data and clock probes. This information is provided to help you set up test connections from the probe pod connections.



CLOCK PROBE

Pin	Signal	Pin	Signal
1	NO CONNECT	14	GND
2	NO CONNECT	15	GND
3	NO CONNECT	16	CHANNEL 7
4	NO CONNECT	17	CHANNEL 6
5	NO CONNECT	18	CHANNEL 5
6	NO CONNECT	19	CHANNEL 4
7	NO CONNECT	20	CHANNEL 3
8	NO CONNECT	21	CHANNEL 2
9	NO CONNECT	22	CHANNEL 1
10	NO CONNECT	23	CHANNEL 0
11	NO CONNECT	24	GND
12	NO CONNECT	25	GND
13	NO CONNECT		

DATA PROBE

Pin	Signal	Pin	Signal
1	GND	14	GND
2	CHANNEL 19	15	CHANNEL 9
3	CHANNEL 18	16	CHANNEL 8
4	CHANNEL 17	17	CHANNEL 7
5	CHANNEL 16	18	CHANNEL 6
6	CHANNEL 15	19	CHANNEL 5
7	GND	20	CHANNEL 4
8	CHANNEL 14	21	CHANNEL 3
9	CHANNEL 13	22	CHANNEL 2
10	CHANNEL 12	23	CHANNEL 1
11	CHANNEL 11	24	CHANNEL 0
12	CHANNEL 10	25	GND
13	GND		

Figure 2-1. General-Purpose Probe Pin Connections

MAKING USER COPIES OF FLEXIBLE DISC SOFTWARE

If your state/software analyzer was shipped with a set of flexible discs containing your software, you should make a duplicate set of discs for your use, and protect the original set that you received with your system. Your original set provides the necessary software to run the state/software analyzer, to operate the system functions, to do disassembly of input data, to run performance verifications, etc. The procedure provided below explains how to make a user set of flexible discs.

When you complete this procedure, you will have one flexible disc containing all your system operating routines and another flexible disc containing all the state/software analyzer routines. These two discs will have all the routines that you need for state/software analysis. Then you can store the master discs you received from HP. Use the master set only when you need to make a new set of user flexible discs.

The system flexible disc that you are going to make will contain the measurement-system routines and the monitor routines.

The state flexible disc that you are going to make will contain the state/software analysis routines, the disassembler routines for the microprocessor(s) you will analyze, the default configuration file, and the preprocessor specification.

When you have completed this procedure, you will be familiar enough with the copying of discs to make any other disc arrangements you desire. You should make a user flexible disc to contain the performance verification procedure.

CAUTION

Take care to protect flexible discs. They can be easily damaged.

To make a user set of flexible discs, proceed as follows:

1. Remove two new blank flexible discs from their containers. Label one of them "SYSTEM" and the other one "STATE". Do not write directly on the flexible discs. Use stick-on labels, if available, or a felt-tip pen. These will become your working set of flexible discs when the formatting is complete.

2. Install the master flexible disc for the operating system (disc #3) in disc drive 0 of your mainframe.

3. Install the new disc labeled "STATE" in disc drive 1.

4. Turn on the LINE power switch. The mainframe will execute the boot-up routines and perform the instrument self-test.

5. Press the following softkeys:

(===ETC===) (===ETC===) (floppy) (utilities) (RETURN)

6. An explanation of the floppy utilities routines will be displayed on the CRT. Press the following keys:

(format) (1) (RETURN)

7. A flexible disc must be formatted prior to use. Formatting will initialize the disc and prepare it to receive information; the system will format your new STATE disc in drive 1. When disc 1 formatting has been completed, open disc drive 1 and remove your new STATE disc.

8. Install your new disc labeled "SYSTEM" in disc drive 1 and press the following keys:

(format) (1) (RETURN)

9. The system will now format your new SYSTEM disc. When disc 1 formatting is complete, press the following keys:

(duplicate) (Ø) (RETURN)

10. The system will make an exact copy of the software contained on disc 0 onto disc 1. When the copy is complete, open both disc drives and remove the discs. Store the master OPERATING SYSTEM flexible disc in a safe place.

11. Install your STATE master flexible disc in disc drive 0 and your user STATE disc in drive 1.

12. Press the following keys:

(duplicate) (Ø) (RETURN)

13. Open disc drive 0 and remove the STATE master flexible disc. Store it in a safe place.

14. Install your user SYSTEM disc in disc drive 0.

15. Press the following keys:

(end) (RETURN)

16. If you are going to make a working copy of your inverse-assembler routine, press the following keys:

(floppy) (sys_gen) (RETURN)

17. Remove the SYSTEM disc from disc drive 0 and install the master flexible disc that contains the inverse-assembler routines into disc drive 0.

18. Press the following keys:

(show) (local) (Ø) (and) (local) (1) (RETURN)

19. The CRT will show a list of the software on your master disassembler flexible disc under disc #0 and STATE disc under disc #1. For each of the routines that you want available for use on your STATE flexible disc, press the following keys:

(copy) <all or module names> (from) (local)

(Ø) (to) (local) (1) (RETURN)

20. Perform the preceding step for each of the modules you want to store on your STATE flexible disc.

21. When all of the modules are copied, open disc drive 0 and remove the master disassembler flexible disc. Store it in a safe place.

22. Now copy the PV software. Remember to format the new flexible disc as outlined in steps 2 through 6. Install the master performance verification flexible disc in disc drive 0 and your new (formatted) user disc in disc drive 1. Press the following keys:

(duplicate) (Ø) (RETURN)

23. Install your new SYSTEM flexible disc in disc drive 0 and your new STATE flexible disc in disc drive 1. Then press the following keys:

(end) (RETURN)

This completes the procedure for making a user set of flexible discs. Your instrument is now ready to perform state/software analysis from your user set.

If you have enough spare flexible discs, you might consider write-protecting the set of flexible discs you just made and making a second user copy using floppy-utilities copy. In this way, you would always have a SYSTEM disc and a STATE disc formatted to your needs. Making future copies of these discs would be easier than using the master discs because you would only be making copies of two discs instead of selective copies of several discs.

Chapter 3
GETTING STARTED

INTRODUCTION

This chapter contains procedures that will help you become familiar with the state/software analyzer. You will learn to enter measurement specifications in the state/software analyzer, and gather data from networks under test. This chapter also explains how to interpret the prompt softkeys, and how to use the utility softkeys and hardkeys.

PROCEDURES

The procedures in this chapter will help you feel at ease with the state/software analyzer. They will show you how to connect the state/software analyzer to a target system and how to run tests. You will do trace measurements and overview measurements if your analyzer has overview capability. You will also modify the default test setup to perform special measurements.

NOTE

These procedures assume that you have installed a 60-channel state/software analyzer in your mainframe and have configured a set of user-software as described in Chapter 2 of this manual. The procedures are written for microprocessor analysis. If you are not analyzing microprocessor activity, you may need to alter the procedures.

CONNECTIONS

1. Connect the CLOCK and DATA POD cables from the analyzer to the appropriate connectors on the general-purpose preprocessor or general-purpose probes.

2. If you are using the general-purpose preprocessor, remove the microprocessor from the target system, and plug in the cable from the general-purpose preprocessor to the interface card microprocessor socket. Then plug in the target microprocessor in the socket on the preprocessor cable. If the target microprocessor cannot be removed, use a test connector (not supplied) to interface the general-purpose preprocessor cable to the target microprocessor.

3. If you are using the general-purpose probes, connect the lines of the probes to the appropriate pins on the target microprocessor. For the purpose of this procedure, connect the probe lines as follows:

a. CLOCK POD, line 0: connect to microprocessor clock line.

b. DATA POD 1, lines 0 through ?: connect to microprocessor address bus (channel 0 to address line 0, channel 1 to address 1, etc).

c. DATA POD 2, lines 0 through ?: connect to microprocessor data bus (channel 0 to data line 0, channel 1 to data 1, etc).

d. DATA POD 3, lines 0 through ?: connect to microprocessor status bus.

4. Connect operating power to the mainframe. This completes the mechanical connections required for state/software analysis of a target system.

5. If your software is on a flexible disc, install that flexible disc in drive 1 on the mainframe front panel. Then install the system flexible disc in drive 0.

NOTE

If you have not already generated user STATE and OPERATING SYSTEM flexible discs, refer to Chapter 2 of this manual.

TURNING ON POWER

1. Turn on the LINE power switch. The associated indicator lamp should light. The mainframe will load the system software automatically. If the message "BOOT FAILURE: DISC IN DRIVE 0 IS NOT SYSTEM DISC" appears on screen, turn off power and swap discs in disc drives. Then turn on LINE power again.

2. Either the (state) or (meas_sys) softkey will be on screen. Use step a or b, as applicable.

a. If (state) is on screen, press it and then press (RETURN).

b. If (meas_sys) is on screen, press it and then press (RETURN). Now press (state<number>) and (RETURN).

3. This completes basic formatting of the state/software analyzer. If you are using the general-purpose preprocessor with a dedicated interface, the state/software analyzer will load itself with a format that is compatible with the target system microprocessor. If you are using general-purpose probes, the state/software analyzer will load itself with the general-purpose format. In either case, the state/software analyzer trace specification will now be on screen.

EXECUTING A TRACE MEASUREMENT

1. Turn on the system under test and start it running a program.

2. Press the (execute) softkey of the state/software analyzer, and then press the (RETURN) key. This begins a trace in the state/software analyzer.

NOTE

> Hardware connections may be quickly checked and verified by performing the Activity Test which is part of the format specification described in Chapter 6.
>
> If "Slow Clock" flashes on the display STATUS line, check the analyzer for a clock input.

3. When the trace is complete, the analyzer CRT will show a list of the states it captured from the target system. The trigger line will be the first state on screen. The default of the trigger specification is "trigger on any_state". Press (execute) and (RETURN) again to see that a new trace can be made immediately.

4. Now choose a specific trigger event and write it down. For example, the first column on screen might be POD1 and it might have states listed in hex values. If so, write down the label POD1 and one of the hex values from the column for use as a trigger in the next trace.

5. Return to the trace specification by pressing the following softkeys:

(show) (tracespec) (RETURN)

6. Enter the state you copied for the trigger by pressing the following keys:

(trigger) (_on_) (POD1) (_=_) 04A3H (RETURN)

↑ your label

↑ your hex value. (Remember the "H" and first character must be numeric.)

7. The trace specification should show your trigger entry under TRIGGER.

8. Now press (execute) and (RETURN). "Waiting for trigger" may flash momentarily on the STATUS line until the trigger state occurs.

9. The display of the trace should show the trigger state you entered on the line labeled "trigger".

10. Now you are going to add some information to the display. Press the following keys:

(display) (modify) (RETURN)

11. This brings your present display format to the command line. Use the (FXW) hardkey to move the blinking cursor across the command line (to the right-hand side, past the end of the command.

NOTE

See how the selections of softkeys change as the cursor moves across the command line. The state/software analyzer always offers the proper choice of softkeys for each position in the command line. This helps you enter the correct command syntax by supplying only the appropriate command names at each level of entry. The <RETURN> prompt on the softkey line indicates that the command is complete enough for the state/software analyzer to accept.

12. Now press the following keys:

(then) (count_abs) (RETURN)

13. Notice that an additional column of information has been added to the display. The "time count abs" column shows the total elapsed time from the trigger state to each state on the display.

NOTE

If your display was already full, this column will not have been added as described above. However, all columns are formatted in memory and may be viewed by pressing the (SHIFT) key along with the (→) and/or the (←) key. This will shift the display to the left or right as needed.

14. Press the (TAB) key on the keyboard to move the cursor across the command line. Stop the cursor under the first letter of the entry you added (the "t" in then). If you tab too far, press (SHIFT) and (TAB) to reverse the cursor direction.

15. Press and hold the (DELETE CHAR) hardkey until your added command is cleared.

16. Press (RETURN). Your display will be as it was before you added the "count" information.

17. You can reformat your display by modifying the content of the command line and pressing (RETURN).

TRIGGERING ON A SEQUENCE

1. Execute a trace by pressing the following keys:

(execute) (RETURN)

2. Write down the label that appears at the top of the first column on the display, and write down three of the values from the first column.

Example:

POD1 ⟵ label

081AH
081BH ⟵ values
081CH

3. Return to the trace specification by pressing the following keys:

(show) (tracespec) (RETURN)

4. Set up a sequence with the information copied in step 2 above. You will use the sequence to trigger a trace in the state/software analyzer. To set up the sequence, press the following keys:

(sequence) (term_num) (1) (find) <LABEL> (_=_)

<VALUE> (RETURN)

(Use label and first value copied in step 2 above.)

(sequence) (term_num) (2) (find) <LABEL> (_=_)

<VALUE> (RETURN)

(Use label and second value copied in step 2 above.)

(sequence) (term_num) (3) (find) <LABEL> (_=_) <VALUE>

(enable) (RETURN)

(Use label and third value copied in step 2 above.)

5. The display on screen should have added the title SEQUENCE along with the terms you entered, followed by the word, "enable". At the beginning of the next trace, the sequence element will check each incoming state. When it finds the state that satisfies your term number 1, it will begin searching for term number 2. When it finds the state that satisfies term number 2, it will search for term number 3. When it finds term number 3, it will switch to the "enable" state.

6. Set up the state/software analyzer to trigger when the sequence you specified is completed (sequence element switches to "enable"). Press the following keys:

(trigger) (on) (sequence) (enable) (RETURN)

7. Begin a trace by pressing (execute) (RETURN).

NOTE

"Waiting for trigger" may appear on the STATUS line. This message will flash until the sequencer element finds the sequence you specified. If "Waiting for trigger" lasts too long, press the (halt) softkey and (RETURN). Then go back to the trace specification (press (show) (tracespec) and (RETURN)). Make sure that you entered the correct values in your sequence, and that you added the "H" to identify each hex value.

MAKING AN OVERVIEW MEASUREMENT

NOTE

Overview measurements are only available in state/software analyzers which have 20-channel acquisition and ranging boards installed as POD 1. If your analyzer does not have the overview capability, the OVERVIEW key will give the message "OVERVIEW Not Installed." In this case, skip this procedure and start the procedure titled "Saving Measurement Setups."

1. Return to the trace specification by pressing the following softkeys:

(show) (tracespec) (RETURN)

2. Activate the overview measurement system by pressing the following keys:

(overview) (on) (POD 1) (RETURN)

↑
A ranging label is required (begins with bit 0 - entirely within pod 1). If no ranging label available, switch to the Format Specification and define one.

3. Set up the overview measurement to recognize and collect information about each one of the three values you recorded earlier. Press the following keys:

(overview) (event) 1 (is value) <???> (RETURN)

↑
first value you recorded for sequence test

(overview) (event) 2 (is value) <???> (RETURN)

↑
second value you recorded for sequence test

(overview) (event) 3 (is value) <???> (RETURN)

↑
third value you recorded for sequence test

4. Begin a measurement in the state/software analyzer. A trace measurement will be made at the same time that the overview measurement is made. Press the following keys:

(execute) (RETURN)

5. The state/software analyzer will show a trace list on screen. Press the following keys to see the results of your overview measurement:

(show) (overview) (RETURN)

6. A label graph is displayed. The overview displays have four types of presentations. To see the overview histogram, press the following keys:

(display) (overview) (histogram) (RETURN)

The histogram is a bar graph showing the portion of program activity that was found to be in each one of the three values you entered. This display only compares the three events you entered to one-another. The display does not account for how much activity occurred outside of the values you specified.

7. Add one more event to include all of the activity outside of the values you specified. Press the following softkeys:

(show) (tracespec) (RETURN) (overview) (event) 4

(is_range) (else) (RETURN)

8. Now execute a new measurement to observe the overview information. Press the following keys:

(execute) (repeat) (RETURN)

9. EVENT_04 on your display shows you the portion of activity that occurred outside of the specifications for events 1 through 3. You can see that the histogram is being repetitively updated with new information.

10. While a measurement is in process, you cannot change specifications. Keys are blanked if they can not be entered. Press (halt) and (RETURN).

SAVING MEASUREMENT SETUPS

1. Press the following keys to save your present instrument setup:

(===ETC===) (configure) (save_in) (A) (RETURN)

2. The state/software analyzer will save the instrument configuration in a file called "A" of the type trace. Note the userid that is active. When the state/software analyzer has finished saving the instrument configuration, the STATUS line will read: Awaiting state command - userid. When this message appears, turn off the LINE power switch.

3. Now turn on the LINE power switch. The instrument will execute the boot-up routines.

4. When the monitor level softkeys appear on screen, press (state), (A), and (RETURN). If the monitor level key is (meas_sys) instead of (state), press (meas_sys) and (RETURN). Then press (state<number>) (A), (RETURN).

5. The state/software analyzer will configure itself according to the setup you stored in file A.

6. Press the following softkeys to default the trace specification:

(==ETC==) (default) (RETURN)

7. Now reconfigure the state/software analyzer without returning to the monitor level softkeys. Press the following keys:

(===ETC===) (configure) (load_from) (A), (RETURN)

8. The state/software analyzer will load itself with the configuration you saved in file A. If you had added the (with_data) softkey to your command when you saved your configuration, the analyzer would also have restored any data captured in memory.

This completes an introduction to the state/software analyzer. You have connected the instrument to a target system and run a few simple tests, both traces and overview measurements. You have modified the measurement specifications to obtain specific information, and then you have stored and recovered your measurement setup using trace files. The remainder of this chapter provides detailed information that will help you obtain greater benefit from the utility offered by the state/software analyzer.

GAINING ACCESS TO THE STATE/SOFTWARE ANALYZER

There are two ways to gain access to the state/software analyzer. The way you use it depends on how many analysis modules are installed in your mainframe (emulation, timing analyzer, additional state/software analyzers, etc.).

The monitor level of softkeys is the level that appears on screen at power up. When the state/software analyzer is the only analysis module in the mainframe, the monitor level of softkeys will show a softkey labeled (state). All you do is press (state) and (RETURN), and the system software will gain access to the state/software analysis module directly.

When two or more analysis modules are installed, one of the softkey choices on the monitor level will be (meas_sys). When you press (meas_sys), the instrument will present the measurement system display. This display has the names of each of the analysis modules in the mainframe on separate softkeys, along with numbers. The numbers identify the slots where the associated analysis modules are installed in the mainframe.

If two state/software analyzers are installed, you will see two softkeys labeled (state) . The numbers beside the names identify the mainframe slots where the state/software analysis control boards are installed. Be sure you select the right state/software analyzer for your measurement.

Figure 3-1 shows the avenue for gaining access to the state/software analyzer when your mainframe contains more than one analysis module. It also shows that you use the show softkey to switch from one display specification to another within the state/software analyzer. For those specifications which have subsets (the format specification and overview displays), the (display) softkey will give you transportation between members of the subset. The (end) softkey is used to exit the state/software analyzer. When you press (end) , the analyzer will save its present measurement setup in a file named "Sdc<slot><HPIB>:HP:trace" (where <slot> is the analyzer control board location, and <HPIB> is the analyzer address if in a cluster installation - <HPIB> = 8 when operating stand-alone). The analyzer will continue any measurement in process after the (end) key is pressed. You can also perform non-analysis operations with the equipment, if desired (editor, assembler, etc.), and view the status of other analysis instruments (timing analyzers, etc.) while your state/software analyzer measurement continues to run. You can reenter the state/software analyzer later and read the results of the measurement you left running when you pressed the (end) softkey.

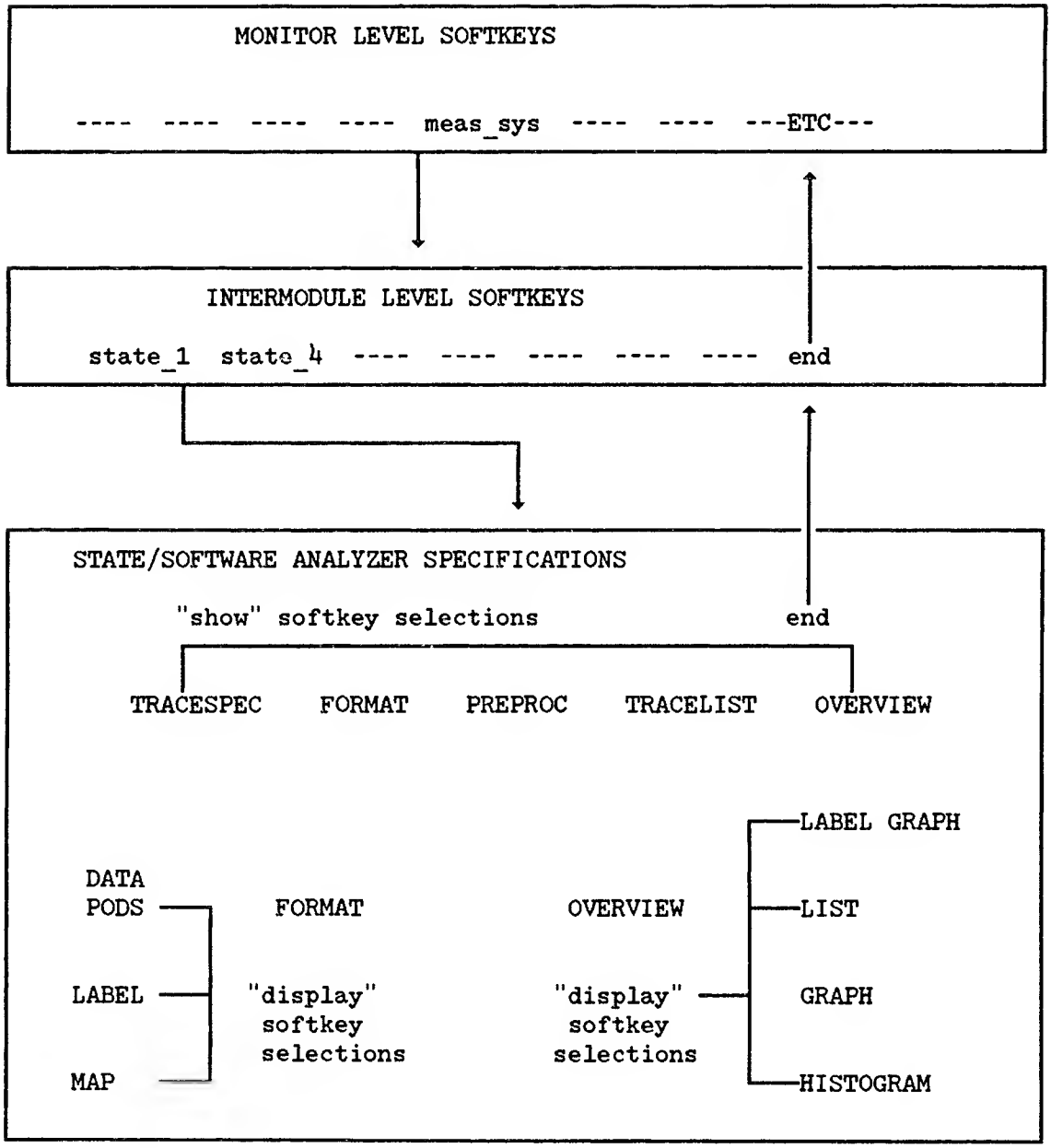


Figure 3-1. Utility Keys Used For Transportation

CONFIGURING THE ANALYZER

There are three measurement configurations which you can have loaded into the analyzer automatically when you first activate it. They are: 1) the default setup, 2) the measurement setup you used last, and 3) any measurement setup stored in memory.

GETTING THE DEFAULT CONFIGURATION

When you gain access to your state/software analyzer by pressing the (state) softkey, the default measurement setup is automatically loaded into the analyzer. At this time, you can press the (execute) softkey and run a test.

The default configuration is dependent upon which preprocessor or probes are connected to the state/software analyzer. If you are using the general-purpose probes, the analyzer will load the configuration in file CDEFAULT:HP:trace. If you are using a dedicated preprocessor, the analyzer will load the configuration for that processor from a file having the following file name C<processor>:HP:trace.

If the default measurement configuration will not make the exact test you want to make, you can modify the setup from the keyboard. Use the softkeys in the format specification and trace specification to set up any unique measurement desired.

You can also change the content of the default configuration file to automatically set up a desired default configuration. Set up the analyzer with the labels, symbols, overview entries, etc., that you want in the default configuration. Then type in (configure) (save_in) C<processor>:HP:trace. This will overwrite the standard default configuration file with your own configuration.

GETTING THE MEASUREMENT CONFIGURATION USED LAST

When you press the (end) softkey, the analyzer stores the present measurement configuration. It will even continue a test if you have a test running when you press (end). To return the state/software analyzer to that setup, press (state), (continue), and (RETURN). If you had a test in process when you pressed the (end) softkey, the analyzer may have finished it and stored your data. If not, your test will still be running when you reenter the state/software analyzer.

NOTE

The (continue) softkey function will not perform the function described above after you have pressed the reset key twice, or after power down, or after running performance verification. You can recover the configuration that was used when the (end) softkey was last pressed by loading file Sdc<slot><HP-IB>:HP.

GETTING A MEASUREMENT CONFIGURATION FROM TRACE FILES

The state/software analyzer can store complete measurement configurations in files. You can develop a library of test setups for your measurement needs and store them in files. The following paragraphs show a procedure for storing and recovering measurement configurations.

a. Set up any desired measurement configuration in your state/software analyzer.

b. Press (configure) (save_in) (A) and (RETURN). This will cause the analyzer to save its present measurement configuration in a file named A under the current USERID.

c. Now you can change the setup any way you like. Your original measurement configuration will still be saved exactly as you stored it in file A.

d. Press (end) to return to the monitor level of softkeys.

e. Press (state) (A) and (RETURN). You will gain access to the state/software analyzer and it will automatically search the files and load the configuration you stored in file A. Use this method when you first enter the state/software analyzer if you want to load a special measurement configuration at the same time.

f. If you are operating the state/software analyzer and you want to load the configuration stored in file A without ending out of the state/software analyzer, press (configure) (load_from) (A) and (RETURN). This will cause the state/software analyzer to change the present measurement setup to the configuration you saved in file A.

g. You can use the (configure) softkey to save as many different measurement configurations as your disc space will allow, each under a different name. Then you can recall and use any of these measurement configurations for later testing. You can also modify the content of any of these configurations once you have them loaded into the state/software analyzer. Simply reload them, make your changes, and then resave them into the same file name.

GETTING A MEASUREMENT CONFIGURATION USING A COMMAND FILE

You can write command files that configure the state/software analyzer and perform automatic measurements. You can include your command file within larger command files that control activities in several instruments. To write a command file for the state/software analyzer, obtain the monitor level of softkeys and enter the Editor Mode, as follows:

(edit) (into) CMDFILE (use any command file name)

When writing your command file, be sure to use the names that appear on the command line for each entry, not the names of the softkeys (use show format_specification not (show) (format)). The parameter-passing feature of command files allows you to include the slot number of the state/software analyzer control board in the command.

An example command file is shown below. It is written to be run from the monitor level.

```
PARMS&SLOTNUMBER

measurement_system

state_&SLOTNUMBER

show map_specification

display map ADDR_MAP

define KEYBOARD range 1000H thru 6000H

show format_specification

define ADDRESS pod_1_bit 0 thru 15 default_map ADDR_MAP

show trace_specification

trigger on ADDRESS = range KEYBOARD

show tracelist

display ADDRESS, POD1, count_absolute, POD2, count_relative

execute

wait 10

show map_specification
```

When you have completed your command file, press the (end) softkey and (RETURN).

To run the command file from the system monitor level, type in the command file name, and press (RETURN). For example:

CMDFILE (not (run) CMDFILE) (RETURN)

When you run the example command file, the display will prompt you for the slot number of the state/software analyzer control board, using the following message:

Define parameter &SLOTNUMBER:

Type the number of the slot where the state/software analyzer control board is installed. Then the system will execute the remainder of the command file.

If the state/software analyzer control board is in slot number 3, for example, you can make the example program run without operator action by replacing the first three lines with the following:

measurement_system

state_3

If the state/software analyzer is the only measurement unit present in the mainframe, replace the first three lines in the example command file with the single line:

state

If you want to run the command file when you are already in the state/software analyzer, delete the first three lines entirely.

If the slot number of the state/software analyzer control board is needed, and your command file does not allow for that parameter, the state/software analyzer will display the message:

syntax error

The state/software analyzer offers special symbols that can be used in place of keywords in command files. These special symbols help you to write shorter command lines.

a. The "," performs the same function as the keyword "then".

b. The ":" performs the same function as the keyword "file".

c. The "#" performs the same function as the keyword "line".

d. The "?" performs the same function as the keyword "global".

ANALYZER SELF-CHECK

The software offers test routines that can be used to assure proper operation of the analyzer hardware. These routines are available at the monitor level of softkeys. To perform self-checking of the analyzer hardware, proceed as follows:

1. At the system monitor level, press the [opt_test] softkey and press (RETURN). The display will show a list of modules that offer Option Performance Verification, and will identify the slots where each is located.

2. Select any 10-MHz analyzer module by pressing the number key on the keyboard corresponding with the Slot # where the module is installed. Then press (RETURN). The top of the display will show "10 MHz State Test: Configuration", and the display will list all of the boards that make up the 10-MHz state/software analyzer.

3. Press the [run] softkey and press (RETURN). The system will perform tests on each of the boards in sequence. It will count the tests, and count the number of test failures (if any).

4. You can [run] tests on all the boards ([all_board]), or test only a single board by typing in its Slot #.

5. You can create a limited list of boards to be tested by using the [include] and [exclude] softkeys, and (RETURN).

6. You can perform a test once (the default mode), or [repeat] the tests until you press "stop", or run tests until a failure is detected ([till_fail]).

7. You can call for a [display] of all the tests available for any one of the boards.

8. You can [list] any test, or test results, to a printer or to a file, or you can [append] your list to an existing file. You can also add a [message] string to be included on your printout or in your file, if desired.

9. Press [end] and (RETURN). This completes the operators self-check. For a detailed description of the performance verification tests, refer to the Service Manual for the state/software analyzer.

Chapter 4
PROGRAMMING MODEL

INTRODUCTION

This chapter is intended to be read after you have done some testing with the state/software analyzer and have developed a working knowledge of how to use it to perform trace measurements and overview measurements with such resources as the sequence and window elements. This chapter provides a model of the analysis capabilities available. The model will help you gain a complete understanding of the state/software analyzer.

This chapter is prepared in a different format from the other chapters of this manual. It consists of a series of five diagrams which focus on aspects of the state/software analyzer. Each diagram leads you step-by-step to an understanding of the architecture of the state/software analyzer. New information on each diagram is highlighted.

BASIC TRACING
(See Figure 4-1.)

The basic tracing diagram shows the elements of the state/software analyzer that are involved in making a trace measurement. Refer to Chapter 8A of this manual for detailed procedures of how to make trace measurements. The preprocessor/probes gather data bits and a clock from the system under test. This information is clocked into the state/software analyzer. The data is delivered to the trace memory and to the state recognition resources.

The state recognition resources examine each state of sampled data to see if it meets the trigger, store, and/or count specifications.

If it meets the trigger specifications, and is the first state to meet the trigger specifications, it is stored in trace memory and identified as the trigger. Then the remainder of memory is filled according to the trigger position specification.

If it meets the store specifications, it is stored in the trace memory immediately following the last state that was stored.

The count specification may require a count of time or a count of states. The counter functions differently, depending on which specification is in force.

If the count specification requires a count of states, each state is checked to see if it meets the specification for the kind of states to be counted. If the state meets the specification, the counter is incremented. If the state also meets the store specification, it is stored in trace memory along with the accumulated count of the counter, and then the counter is reset to zero. At the end of the measurement, each state stored in trace memory will have a corresponding count that identifies the number of states transacted between each stored state.

If the count specification requires a time measurement, the counter will measure time. Each time a new state is stored in memory, the present time measurement from the counter will be stored in memory and the counter will be reset to zero.

The display function of the state/software analyzer will read the content of the trace memory and format trace lists and/or overview label graphs of your choice and display these on the CRT.

TRACE WITH SEQUENCER

ENABLES AND DATA INTERPRETATION

(See Figure 4-2.)

This diagram repeats the trace measurement functions that were shown on Figure 4-1, and shows how these use the sequence and window elements. Refer to Chapter 8B of this manual for a detailed discussion of how to specify operation of the sequence and window elements.

The strobed input data is delivered to the sequencer where it is examined by the three sequencer elements. If the sequence and/or a window element finds that the sampled state meets its enable or disable specification, the element will switch its output accordingly. The three sequencer outputs are delivered to the trace measurement operation in the analyzer where they are available for enabling the trigger, store, and count specifications.

The following commands include the sequencer elements in the specifications for the trigger and store functions:

(trigger) (enable) (sequence) (RETURN)

(store) (enable) (window) (one) (RETURN)

These two commands allow the state/software analyzer to recognize its trigger only when the sequence element is in the "enable" state, and allows storage of states only when window one is in the "enable" state.

The outputs of the three sequencer elements are also delivered to the trace memory where they are stored along with state and count information. You can obtain lists of memory content that show the sequence and window conditions that were in effect when each state was captured in memory.

This diagram also shows additional detail of how the display function interprets the information in the trace memory. The labels and symbols in your format specification can be used to identify the trace memory content. The display function can also process the trace memory content through an inverse assembler to list captured information in the mnemonics of the system under test.

THE OVERVIEW MEASUREMENT

(See Figure 4-3.)

The basic overview diagram shows the functions of the state/software analyzer that are involved in making overview measurements of activity derived from a ranging label. Refer to Chapter 8C of this manual for details of how to make overview measurements.

The preprocessor/probes gather data bits and a clock from the system under test. Data is strobed into the analyzer according to the clock specification, in the same manner as for trace measurements. The overview functions collect only the states present on the selected ranging label. The states are delivered to the range/overview event decode. This function classifies each state according to the overview event specifications you entered before the start of the measurement. The code for each event classification is sent to overview memory.

The state/software analyzer display function reads the event codes in overview memory and formulates histograms, lists, and graphs from the collected events.

THE SEQUENCE OVERVIEW MEASUREMENT

(See Figure 4-4.)

This diagram shows basic overview functions from Figure 4-3, and adds the state/software analyzer functions used to obtain the full overview measurement capabilities. Three modes of overview measurements can be made with the state/software analyzer: time-count, state-count, and ranging-label overview. This diagram shows the state/software analyzer elements that are involved in making all three modes of measurements. For further information, refer to the chapters of this manual that describe how to use the sequence/window elements and how to make overview measurements (chapters 8B and 8C, respectively).

Ranging Label Overview. The state/software analyzer derives a series of states from a set of leads identified by a ranging label. Each of these states is classified. The classifications are stored in overview memory. The functions involved in this measurement are the data strobing which obtains the states, the range/overview event decode which detects the corresponding event classifications, and the overview memory which stores the event classifications. This mode of overview measurement was discussed for Figure 4-3. Ranging label overview can also be windowed or sampled by controlling overview enable from a sequence or window element.

Time-Count Overview. This measurement classifies periods of time that elapse between the occurrence of one selected state and another selected state in program flow. The states between which you overview time are entered into the specification for the sequence element or one of the window elements. Set up the element to enable when it finds the start-state and disable when it finds the end-state. Then the overview event detector classifies the length of each enable period from the selected sequence element. The classifications for each event are stored in overview memory.

You can add another sequencer element to the specification to window the analyzer so that only time during selected types of program execution is included in the classification. Set up one of the other sequencer elements to enable at the start and disable at the end of a selected subroutine. Use the output of this sequencer element to enable the overview counter so that it measures only the time spent executing the subroutine during the enable period.

State-Count Overview. This measurement classifies counts of states that occur between one selected state and another. It is similar to time-count overview. The start-count and end-count states are entered into the specification for the sequence or one of the window elements.

During each enable period, the state/software analyzer counts the states it detects. At the end of each enable period, the count is classified by the range/overview event decode, and the classification is stored in the overview event memory.

As with the time-count overview, you can use a second sequence or window element to enable the count of overview events.

In the state-count mode, you can have an additional level of qualification for your overview event counts. Instead of counting all states, you can have the overview function count only the occurrences of selected states. (You might overview counts of the number of times that address 03FH occurs during I/O reads.)

COMPLETE ANALYZER PROGRAMMING MODEL

(See Figure 4-5.)

Figure 4-5 shows the entire state/software analyzer programming model. It includes all of the functions that were shown in figures 4-1 through 4-4, and additional capabilities of the state/software analyzer. These additional capabilities are discussed in the following paragraphs.

Overview Trigger. The overview function can trigger a trace measurement. If you enter a trigger specification such as "trigger on overview_event 1", then the trace measurement will be triggered when the range/overview event decode finds the first state that meets the specification of overview event 1. You can not use an internal trigger enable specification when triggering on the occurrence of an overview event. You can, however, receive a trigger enable from another analyzer on the intermodule bus, or you can send a trigger enable from the state/software analyzer to other analysis modules on the intermodule bus.

Master Function. This function provides a higher level of windowing. Refer to Chapter 8F for details. The master function enables or disables the entire state/software analyzer. When master is in the "enable" state, each of the individual state/software analyzer functions is free to operate at its specifications. When master is in the "disable" state, only the sequence or window element that is used by the master function is free to operate. The remaining functions of the state/software analyzer are suspended until master returns to the "enable" state.

IMB. The IMB (intermodule bus) carries five control signals to and from the other analysis modules in the mainframe. These signals enable synchronization of measurements involving two or more analysis modules. Each signal carried by the IMB is described in the following paragraphs.

(1). The master function is carried by the IMB. It synchronizes measurement execution and halting and can suspend operation in all the associated analysis modules when measurements include software windowing.

(2). The delay clock is generated by the state/software analyzer. It can be used by the other analysis modules to delay taking their measurements until after a selected number of state clocks following trigger recognition.

(3). The trigger line can carry simultaneous trigger recognition from the state/software analyzer to the other analysis modules, or from one of the other analysis modules to the state/software analyzer, or perform OR'd triggering between two or more modules.

(4). The storage enable can be received from some other analyzer, or it can be sent by the state/software analyzer to window storage in all of the receiving analysis modules.

(5). The trigger enable can be generated by the state/software analyzer when the sequence or a window element switches to enable, or when the state/software analyzer finds its trigger or trace-point. The trigger enable can also be received from some other analysis module on the intermodule bus.

Assert Functions. The assert functions that the state/software analyzer can provide are halt and stimulus. The state/software analyzer can send these functions to the preprocessor, or to BNC connectors on the rear of the mainframe, or both. The stimulus function generates a pulse when trigger is recognized or when selected events occur in one of the sequence elements. The pulse can be used to simulate interrupts in the system under test. The halt line carries a logic level which can be used to halt operation in the system under test. The halt can be generated after the occurrence of the trace point in program flow, or when the state/software analyzer completes a trace measurement.

Preprocessor Configuration. When a dedicated preprocessor is connected to the state/software analyzer, it identifies itself during power-up. This identification calls forth a corresponding operating configuration. This configuration sets up the format specification in the state/software analyzer to be compatible with the type of system identified by the preprocessor. Refer to Chapter 9 and the separate preprocessor reference manual for details.

Sequencer Events And Terms. This bus carries the pulses that are generated when events are recognized in any of the three sequencer elements. Each window element generates a pulse when it recognizes its enable or disable event. The sequence element generates pulses during these event recognitions, and whenever any terms within its specified sequences are found. These pulses can be used to qualify trigger and store specifications for trace measurements. They can also be used to generate stimulus pulses. Refer to Chapter 8B for details of how to use the sequencer elements.

Ranging Tradeoffs. Ranging in the state/software analyzer is done by the range/overview event decode. Ranging labels are labels assigned to contiguous sets of probe leads beginning with line 0 and entirely contained in pod 1, when pod 1 is a ranging pod. The state/software analyzer offers two ways to use ranging labels: (1). specifying ranges of states to be recognized as parameters for trace measurements, and (2). specifying overview measurements as described in Figure 4-3. If you use state/software analyzer ranging to accomplish one of these activities, ranging cannot be used for the other.

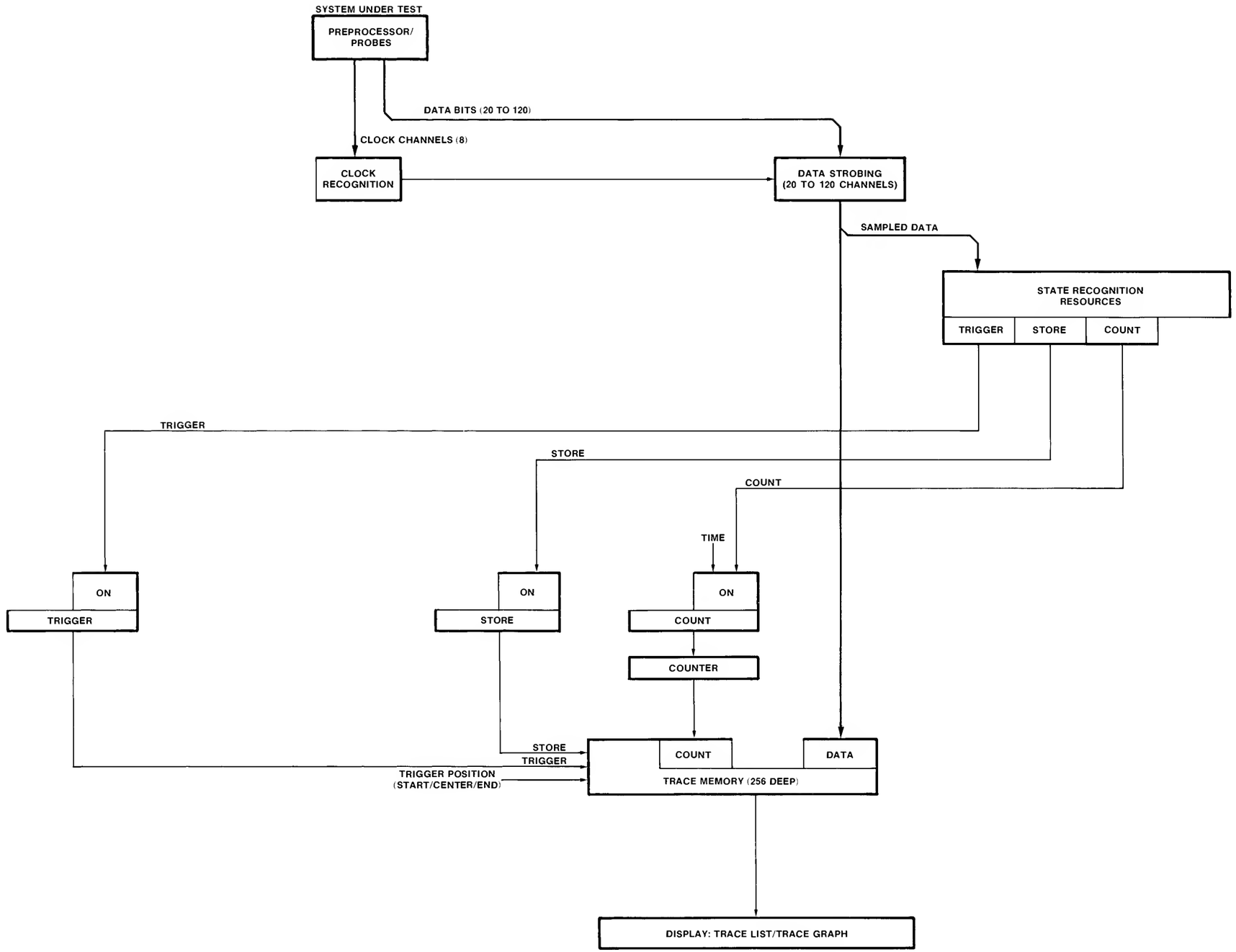


Figure 4-1.
Basic Tracing

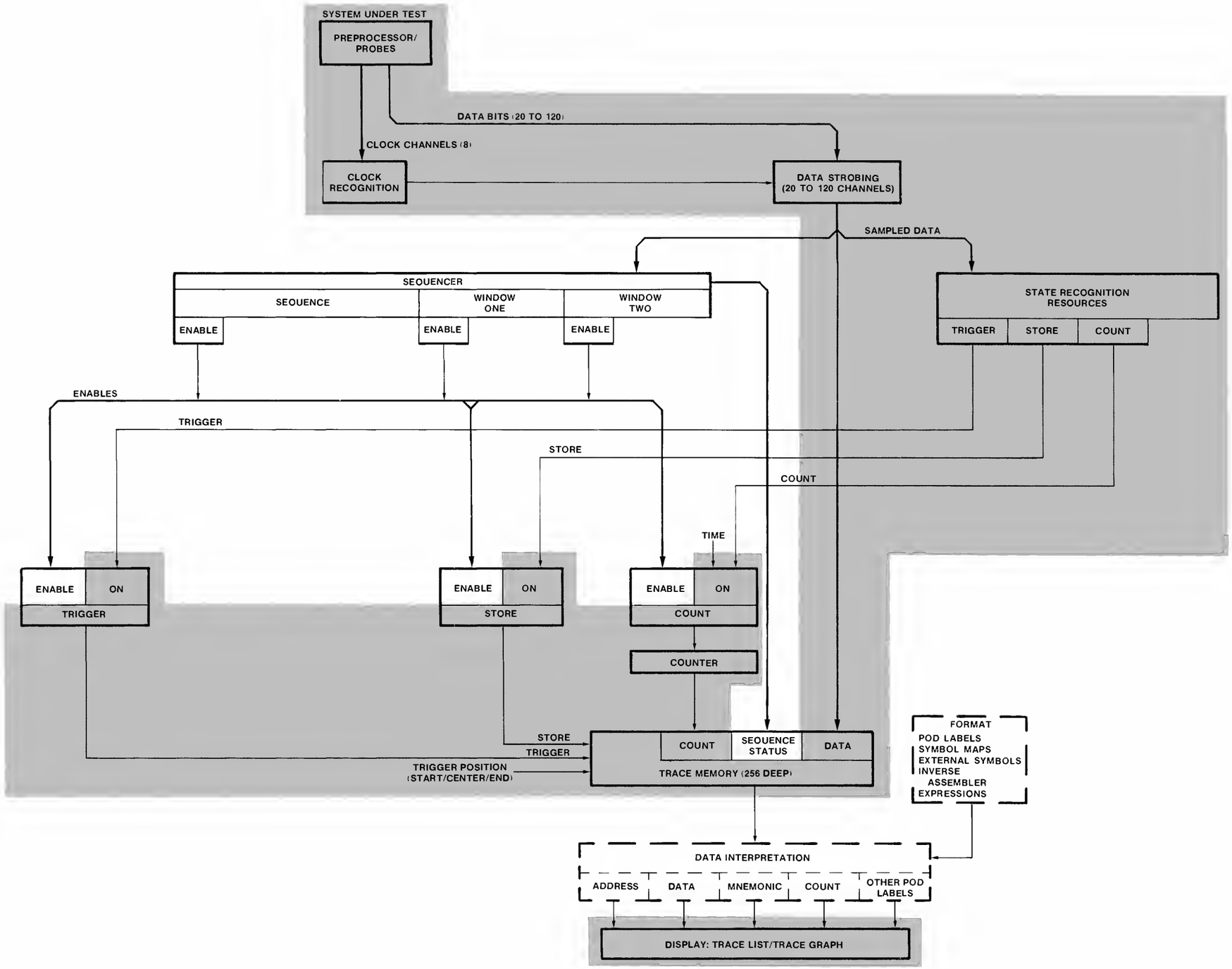


Figure 4-2.
Tracing With Sequence Enables

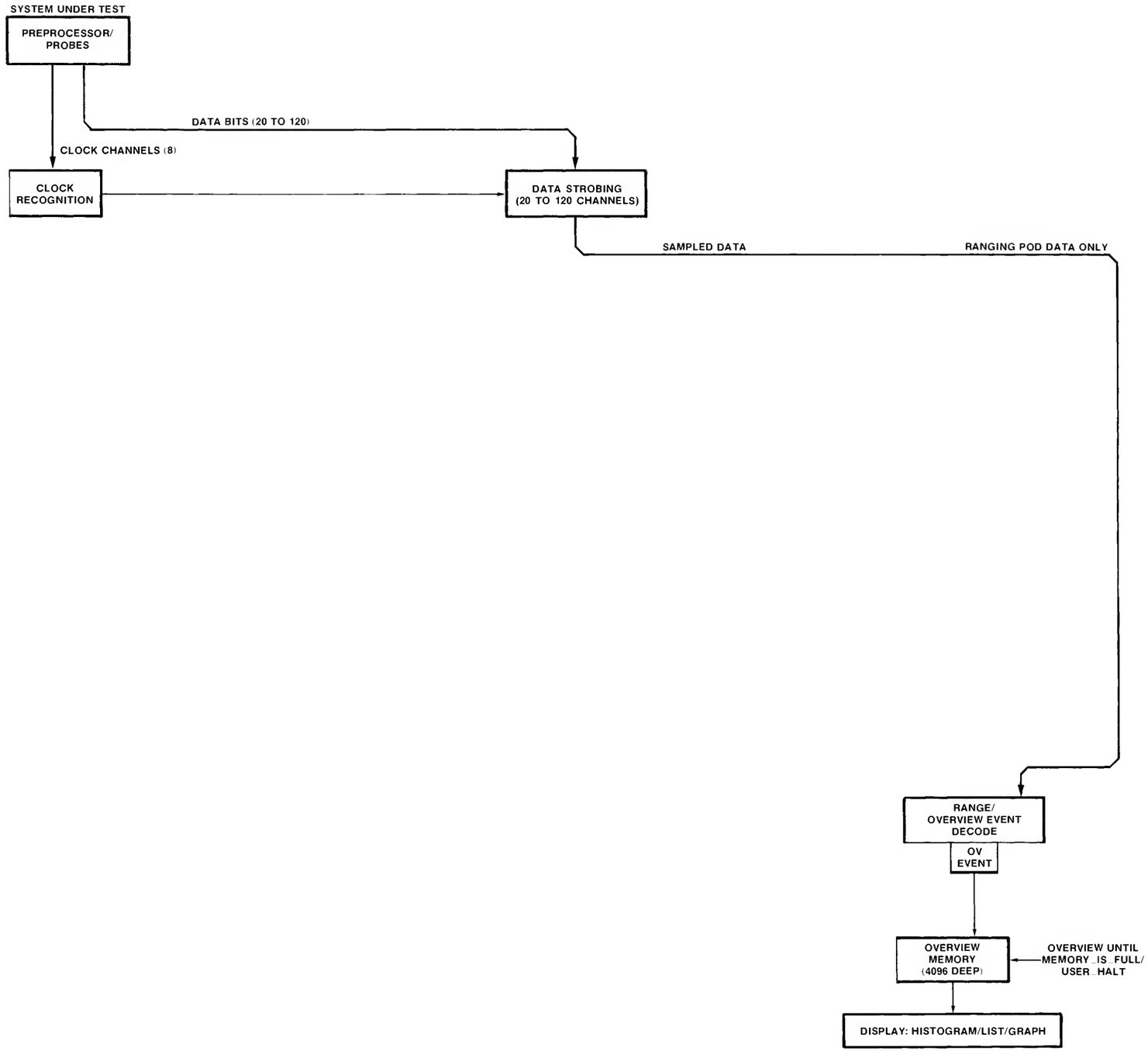


Figure 4-3. Basic Overview

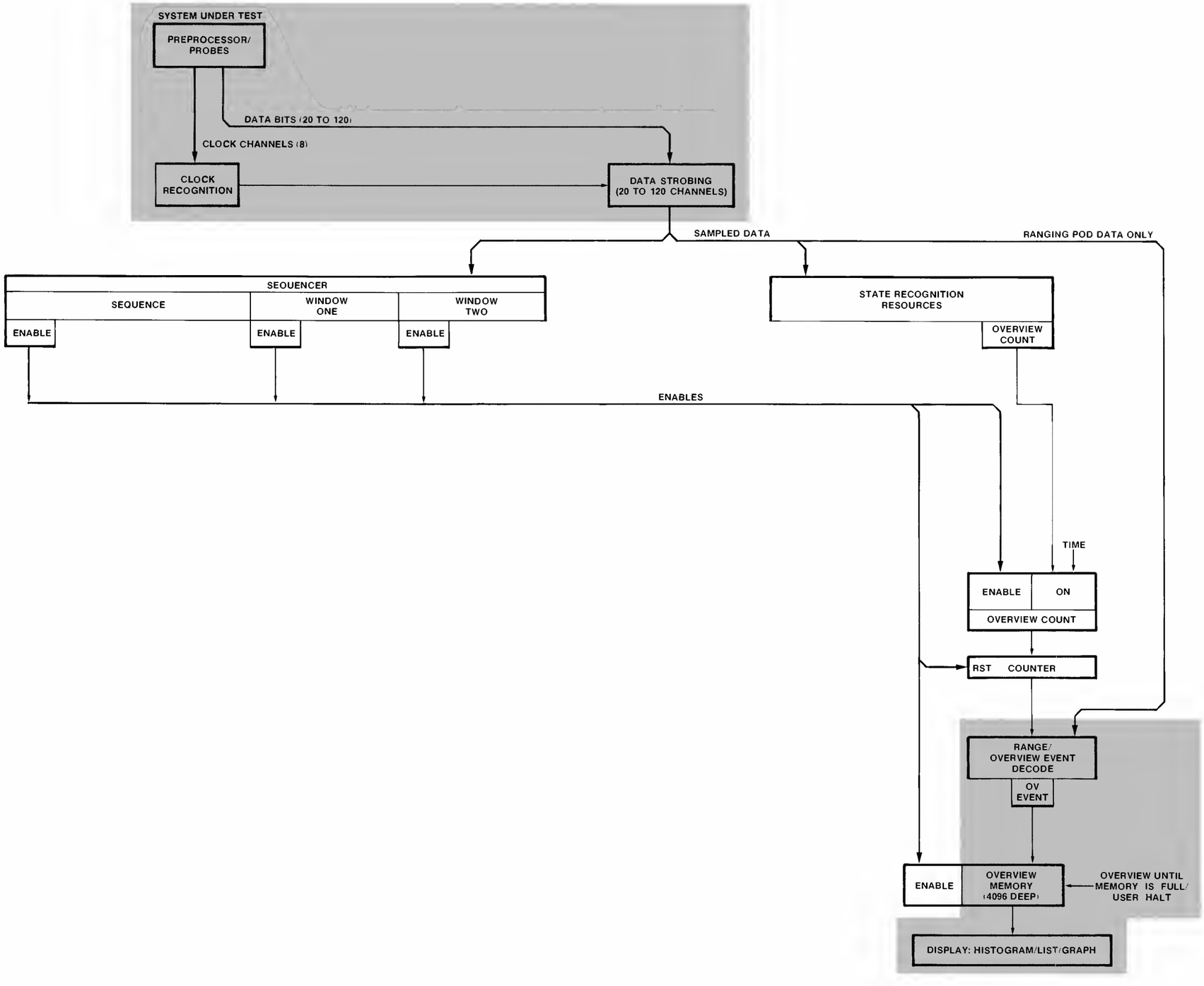


Figure 4-4.
Overview With Sequence Enables

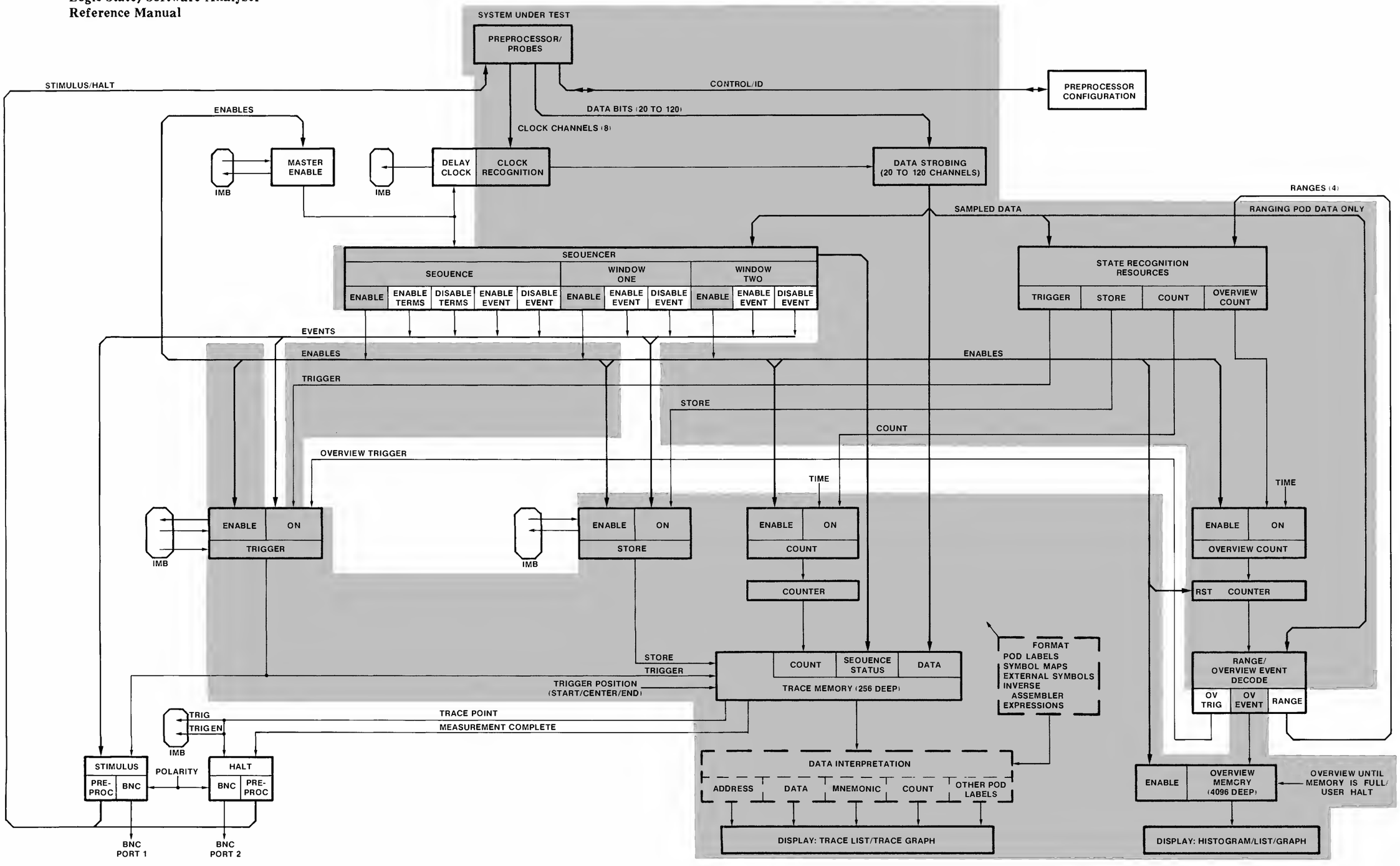


Figure 4-5.
Complete State/Software Analyzer Programming Model

Chapter 5
KEYBOARD OPERATION

INTRODUCTION

This chapter discusses how to use the keyboard to make entries for operating the state/software analyzer, and how the state/software analyzer directs the entries you make. Procedures are described for entering numbers from the keyboard. This chapter also shows you how to use the special keyboard character for adding comments in your command files. Finally, the utility keys are discussed: both the utility softkeys and softkey prompts, and the utility keyboard keys.

DIRECTED SYNTAX

The state/software analyzer places a row of softkey names across the bottom line of the CRT face. These softkey names identify the functions to be obtained by pressing corresponding keys in the row at the top of the keyboard. When you press one of the softkeys (selecting a parameter), the names of all the sofkeys change. The new row of softkey names offer selections that can be made following the previous softkey entry.

By directing the syntax of your entries, the state/software analyzer minimizes syntax errors. The line of softkeys always identifies the appropriate entries to be made at any point while you are formulating a command.

ENTERING NUMERIC VALUES INTO THE STATE/SOFTWARE ANALYZER

You can enter numbers into the state/software analyzer using any of the four standard number bases. Place the applicable letter symbol (B, O or Q, D, H) at the end of your number to define its base. Refer to the following examples:

```
1000B = 1000 binary
10000 or 1000Q = 1000 octal
1000H = 1000 hexadecimal
1000D or 1000 = 1000 decimal
```

NOTE

Decimal is assumed if you do not specify the base when you enter a number.

Hexadecimal numbers must begin with a numeric symbol. Example: 3FAH, 0FFH, but not F44H.

ENTERING COMMENTS IN COMMAND FILES

The state/software analyzer can process a special symbol which you can use in your command files when you want to introduce a comment. The symbol is the semicolon ";". The analyzer will not read any material following a ";" in any line of your command file. It will start loading new instructions only after it finds the next carriage return.

UTILITY SOFTKEYS

The utility softkeys are available in all of the display specifications. They allow you to [execute] and [halt] traces, to [show] any desired primary specification on screen, to [display] any specification which is a subset of one of the primary specifications, to save and reload files that [configure] the state/software analyzer, to obtain a [copy] of your test activity, and [end] your session on the state/software analyzer without losing your specifications or stopping measurements in progress. You can use these keys to enter the name of the [absolute] file under test, and to enter [wait] commands for control in command files. You can also [calculate] values. The utility softkeys are described here, in detail.

execute

When you can press the [execute] softkey, the state/software analyzer will gather data according to its present measurement configuration. If you have entered specifications for a measurement, the analyzer will run a trace according to your specifications. If you have not entered specifications, the analyzer will run a test according to the resident default configuration. If an intermodule bus measurement is set up, each of the analysis modules involved in the measurement will be started. During a measurement, this softkey changes to [halt].

halt

When you can press the [halt] softkey, the state/software analyzer will stop any measurement in progress. If an intermodule bus measurement is in process, all modules involved in the measurement will be halted.

show

When you press the [show] softkey, the analyzer will offer you access to all of the primary specifications (trace_specification, format_specification, map_specification, trace_list, overview displays, and preprocessor_specification). Use the [show] softkey for transportation from one primary specification to another.

display

The [display] softkey appears in the primary specifications that have display subsets: the format_specification, the map_specification, and the overview displays. The [display] softkey is used for transportation from one member of the subset to another. For example, the [display] softkey selects between the data pods, data labels, and symbol maps displays when you are in the map_specification.

end

The [end] softkey is used for transportation from the state/software analyzer to the measurement system display (or system monitor if only one analysis module is present). When you press the [end] softkey, the system will exit the state/software analyzer and return to either the measurement_system display or monitor level display. When you leave the analyzer by use of the [end] softkey, your measurement configuration will be stored in state/software analyzer file Sdc<slot><HPIB or 8>:HP:trace. If your state/software analyzer has a test in progress, it will continue to run that test without interruption after you press [end] . Your data will be collected automatically. You can reenter the state/software analysis module at any time and observe the results of your test by pressing the [state] and [continue] softkeys, and the [RETURN] key.

NOTE

If you do not include the [continue] statement in your command, your present measurement configuration will be lost along with all data collected when you next enter the state/software analyzer. However, you can recover the previous measurement configuration by using [configure] [load_from] Sdc<slot><HPIB or 8>:HP [RETURN].

absolute

The [absolute] softkey is used to enter the name of the absolute file to be analyzed, and it commands a check (and possible construction or updating) of the database for that absolute file. See Figure 5-1.

NOTE

The database is constructed by the state/software analyzer. It consists of the content of the link_symbols file for the absolute file and the assembly_symbols files for each of the relocatable modules linked in the absolute file.

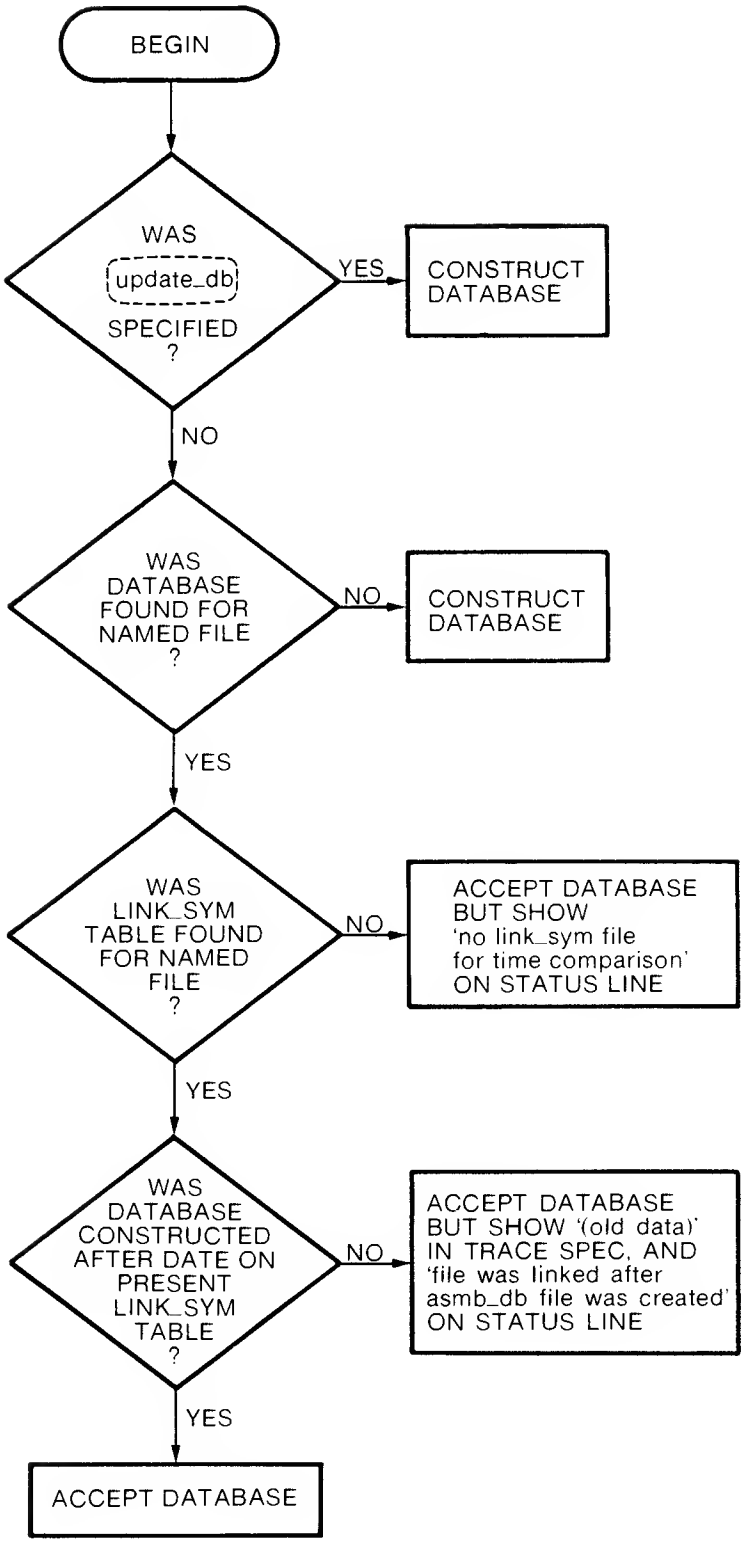


Figure 5-1. Absolute_is <FILE> Update_database Logic

copy

You can press the (copy) softkey at any time to obtain a copy of the content of your display or any specification on a printer, or to store the selected specification or contents of your display in file memory. The (enhanced) softkey causes the state/software analyzer to generate special printing characters to obtain the displayed underlines on your copy.

configure

The (configure) softkey is used to store new measurement configurations in trace files, and to recover existing measurement configurations from trace files.

a. The command (configure) (save_in) <FILE:ID> will cause the system to save the present measurement configuration in memory under the file name and userid that you specify. You can add (write_protect) to this command line if you want to ensure that your file cannot be accidently overwritten. If you specify write_protect for your file, you will not be able to save another configuration by that name until you purge that file using the monitor level softkeys.

b. The command (configure) (load_from) <FILE:ID> will cause the system to load the measurement setup stored in that file and data, if present, into the state/software analyzer.

calculate

The (calculate) softkey is available in all the specification displays, the list display, and the overview displays for making general-purpose calculations. You can use the (calculate) softkey to convert absolute address values to relocatable address values to help identify locations in your listings. You can also obtain the values of map symbols.

Examples:

(calculate) MAP_1.FRAMUS (RETURN) The value of the symbol FRAMUS on MAP_1 will be shown on the STATUS line.

(calculate) MAP_1.START (+) 5 * MAP_2.CHARS (=) 37Q (RETURN). The value of the symbol START from MAP_1 plus 5 (decimal) multiplied by the value of the symbol CHARS from MAP_2 plus 37 (octal) will be shown on the STATUS line.

wait

The [wait] softkey is useful for entering "wait" statements when you are preparing command files. You can have the wait statement in effect until the analyzer completes a measurement, for any period (in seconds) you define, or until any keyboard key is pressed. The wait command can be overridden at any time by pressing a keyboard key.

"wait measurement_complete" ensures completion of measurement before the next command file line is accepted.

"wait 10" ensures a 10-second period will elapse for holding the present display on screen before the next command file line is accepted.

"wait" stops the command file until you press a key on the keyboard.

PROMPT SOFTKEYS

Prompt softkeys are softkey names in capital letters enclosed in angle-brackets "<>". All of the prompt softkeys are listed and described in the glossary at the end of this manual. The four most common prompt softkeys are also discussed in this paragraph. If you press a prompt softkey, the STATUS line of the display will explain the meaning of the prompt.

<RETURN>

When this softkey appears on the display, it indicates that the command line you have entered is a complete command. This does not mean that you have to stop adding to your specification, but you can if you want to. If the <RETURN> softkey is not on screen, it indicates that your present specification is not complete, and the analyzer will not accept it as entered.

<PATTERN>

This prompt indicates that you can type in a number which contains don't care characters (X's) so that you can define a range of values. If the number you type in is larger than the label width, the analyzer will automatically delete the most significant bits.

<VALUE>

This prompt indicates that you can type in a number up to 32 bits wide. It can not contain don't care characters. If the number you type in is larger than the label width, the analyzer will automatically delete the most significant bits.

<INVALID>

When this softkey name appears on screen, it indicates that the state/software analyzer has detected an invalid entry in the command.

UTILITY KEYBOARD KEYS

The following keyboard keys are useful when you are making entries in the command lines of state/software analyzer displays.

RECALL

The (RECALL) key will cause the state/software analyzer to show the preceding command line on screen. The state/software analyzer has a command line memory which the (RECALL) key activates. Each time you press (RECALL), the state/software analyzer steps one execution further back into its memory of command lines.

TAB

The (TAB) key moves the cursor rapidly through the command line on screen. This key is useful when you are making modifications to long specifications. By pressing (TAB) , you step the cursor from entry to entry forward through the specification. By pressing (SHIFT) and then (TAB) , you step the cursor backwards through the specification.

INSERT/DELETE

The (INSERT CHAR) and (DELETE CHAR) keys are used to edit the content of a command line. The (INSERT CHAR) key will open a space before the present position of the cursor so you can add entries in the command line without losing the entire specification. When you press the (DELETE CHAR) key, the entry directly at the cursor will be eliminated and the remainder of the command line will shift left.

Chapter 6
THE FORMAT SPECIFICATION

INTRODUCTION

The Format Specification is composed of two displays: the data_pods display and the data label displays (one for each label). You enter the format specification by pressing the (show) and (format) softkeys, and pressing (RETURN) . Obtain one of the two displays of the format specification by pressing the (display) softkey along with either (data_pods) or (label) and (RETURN). Use these displays to define the labels and clocks that the analyzer uses to perform data acquisition. The labels identify each data input or set of data inputs to be monitored. The clock is identified by a variety of activities found on up to eight clock probe lines.

If you are using one of the dedicated preprocessors as a probe, the analyzer will set up your format specification with the data labels and clock identifiers normally used by the associated microprocessor. This will be done automatically during entry to the state/software analyzer (after pressing the (state) softkey).

If you are using the general-purpose probes, the format specification will be set up with a default configuration which you can modify to correspond to the system you are monitoring.

This chapter will help you set up the displays of the format specification when you are using the general-purpose probes. It will also help you add labels and modify the standard setup provided when you use a dedicated preprocessor.

DEFINITIONS OF TERMS

Four terms used in this manual have unique definitions when you are performing state/software analysis: label, symbol, map, and default_map. You must understand these four terms before you can comfortably set up the state/software analyzer and use its capabilities in measurements. Symbol and map are defined in Chapter 7 (The Map Specification). Label and default_map are defined in the following paragraphs:

LABEL

A label is the name you assign to a set of probe leads. For example, you may have bits 0 through 15 of pod 1 connected to the address bus of your target microprocessor. In the format specification, you can assign the label ADDRESS to those sixteen lines, and that label will then be available to use in all of the displays where you might want to identify those lines. The analyzer will use that label in its displays when it

shows you the activity collected from those lines. It will also present that label to you when you are setting up a test in the trace specification.

Labels can be composed of from 1 to 16 upper-case and/or lower-case letters, numbers, and underscores in any combination, except the first character must be an alpha character.

DEFAULT_MAP

The default_map entry is used to assign a symbol map to be used by a label. After this entry, the state/software analyzer will automatically use symbols from the default map in all displays. The way you make this entry is with the following softkeys:

(define) (ADDRESS) (def_map) (ADDR_MAP) (RETURN)

This entry instructs the analyzer to default to the symbols on the map named ADDR_MAP whenever activity is specified under the label ADDRESS. You can also enter absolute in place of a map name, if desired. In this case, the analyzer will only accept and display values in absolute numbers in reference to that label.

THE data_pods DISPLAY

In the data_pods display, you can create labels for each of the incoming lines from the probe pods to identify the activity that will be carried by them. You can label single lines (such as a single flag function), and you can label groups of lines (such as all the lines in the address bus under the label ADDRESS).

You can also create overlapping labels. You can create a label called STATUS which identifies all activity on the status lines, and another label called READ_WRT which identifies just the activity on the read/write line in the status bus. The activity measured under each of these labels can be displayed on trace lists. You can specify trigger, count, or store activity on either of these labels, but you cannot create a specification that AND's these two labels together. If you specify trigger on a certain condition in the STATUS label, the analyzer will not allow you to AND activity under the READ_WRT label because the STATUS label already includes the value of the READ_WRT label.

You can group lines that are obtained from more than one pod, and the analyzer will gather the activity from all those lines and interpret them under the assigned label. For example, you can include lines 17, 18, and 19 from pod 1 and lines 0, 1, and 2 from pod 2 all under a single label because all of these lines are contiguous. Be careful if you are setting up labels on two or more pods which use different threshold levels.

You cannot label a group of lines unless they are contiguous. For example, you cannot label lines 4, 5, 6, 8, and 9 of pod 1 because line 7 is missing.

You cannot define a label name that is also an analyzer keyword (such as a label called 'count' because it is a keyword in the trace specification). You cannot define a label having the same name as a map (such as a label called OPCODE when you also have a map named OPCODE).

When you have many labels defined and/or you have more than three data probe pods, the display may not show all of your assignments. Use the (ROLL UP) and (ROLL DOWN) keys along with the shift-left and shift-right arrow keys to roll the display window.

An (activity) softkey places a line labeled "activity test" on screen under the pod bit numbers. The 'H' or 'L' under each bit shows the state of each line (changing 'H' and 'L' indicates an active line). Use the "activity test" as a first level of performance verification and to check your probe pod connections.

See Figure 6-1 when reading the following instructions. referenced to the sample display shown in Figure 6-1.

DEFINING A NEW DATA LABEL

Define a new label by pressing the (define) softkey. Now type in the name of the label that you want to create. You can enter a label name composed of from 1 to 16 upper-case or lower-case alpha and/or numeric characters plus underscores, beginning with an alpha character.

Press the softkey that names the lowest numbered pod to be included in your labeled set, and type in the number of the lowest numbered bit to be used in that pod.

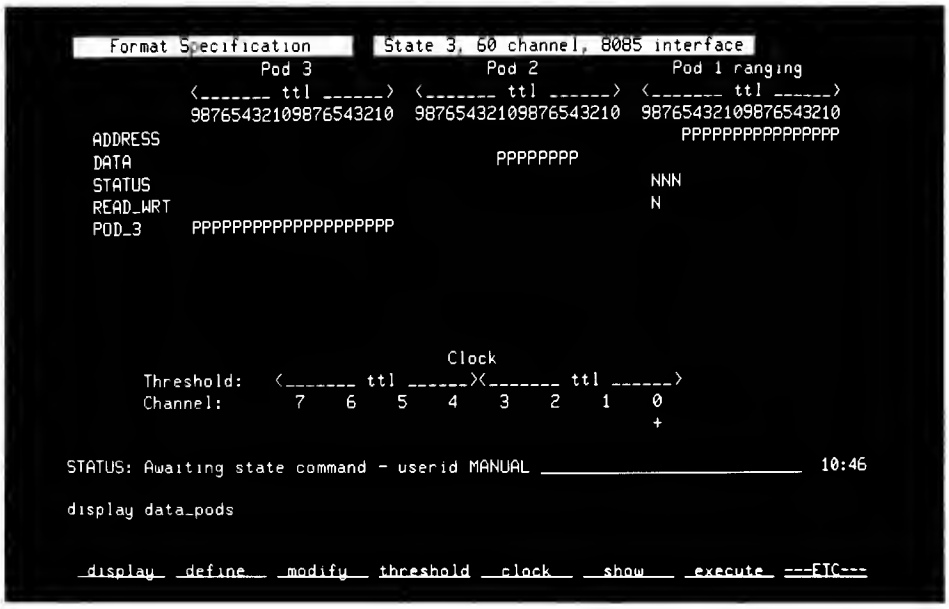
At this point, the <RETURN> prompt will indicate you have a specification the analyzer will accept. You can press the (RETURN) key and the analyzer will accept your label, and assign it to the bit you specified.

If there are two or more bits to be included in your label, you can either use the (width) softkey and enter the total number of bits, or use the (thru) softkey and enter the pod number and bit number of the highest bit to be included in your label. You can include up to 32 bits in any label.

You can specify the logic polarity of your labeled probes by pressing the (polarity) softkey and either (positive) or (negative) . You can also let the analyzer select the default polarity (positive true).

You can assign your label to default to the symbols on a symbol map by pressing (def_map) and the softkey that carries the name of the map. Without this selection, the analyzer will use absolute values to accept data entry and format displays.

Press the (RETURN) key. Your label will appear on the display along with "P" or "N" under each bit included in the labeled set. The "P" indicates positive logic. The "N" indicates negative logic. You can also use the (rename), (delete), and (modify) softkeys along with the keys described above to edit your label.



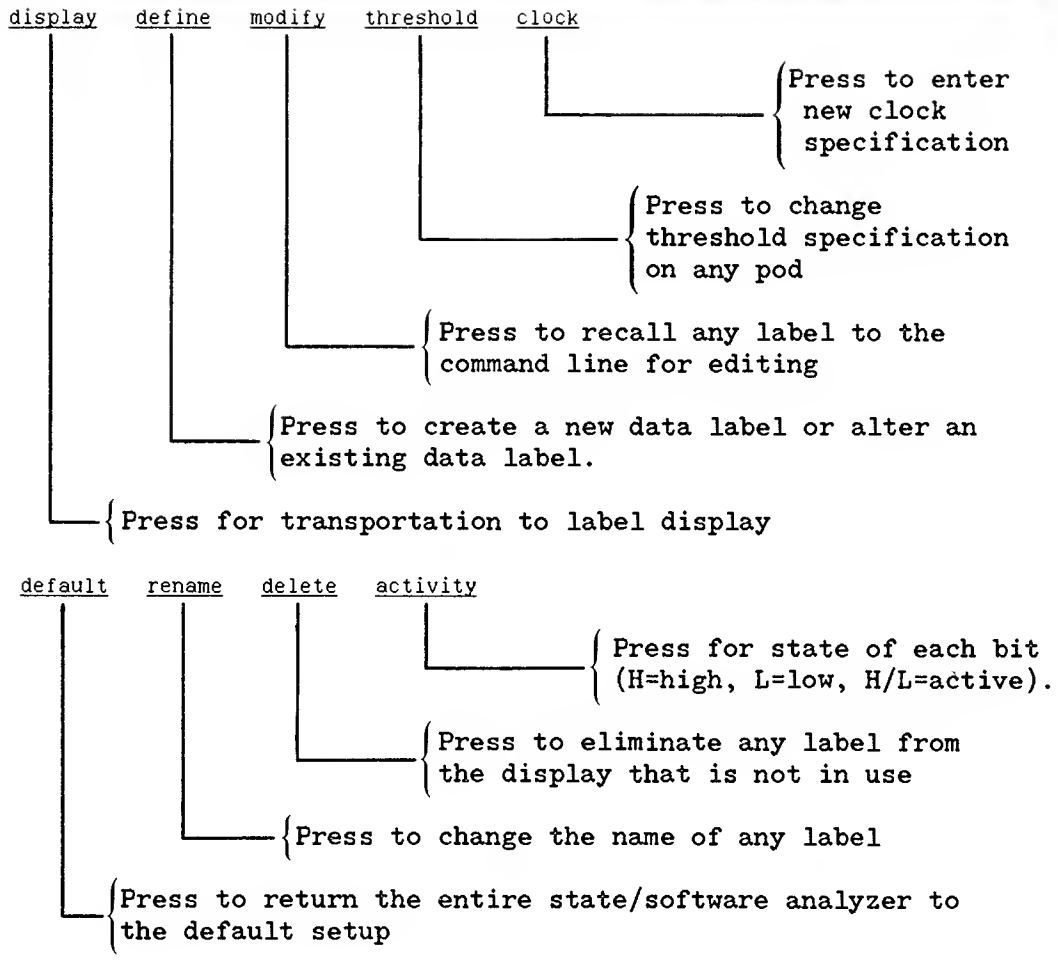


Figure 6-1. The data__pods Display

MAKING A NEW CLOCK ASSIGNMENT

There are eight clock lines which can be used to make up a condition on which to clock the incoming data. You can use any one of these lines to carry the clock or you can use any number of them (up to all eight) to both qualify and clock your incoming data. Any specification that you make to any line will be identified under the appropriate channel number as follows:

a. The plus sign will appear below the channel number if a rising edge in that channel will satisfy the clock requirement.

b. The minus sign will appear below the channel number if a falling edge in that channel will satisfy the clock requirement.

c. The plus/minus sign will appear below the channel number if either edge can satisfy the clock requirement when found in that channel.

d. The letter H will appear below the number if a high state must be present in the associated channel when the clock edge(s) is found.

e. The letter L will appear below the number if a low state must be present in the associated channel when the clock edge(s) is found.

Begin your clock specification by pressing the (clock) softkey.

You can have the analyzer clock on either a rising edge, falling edge, or both rising and falling edges. Press either the (rising), (falling), or (both) softkey.

Now press the softkey that identifies the channel where that clock edge is to be found.

You can press the (RETURN) key and the analyzer will accept your clock specification as entered. If you need a more complicated clock specification, press either the (or) or the (and) softkey.

An "or" entry allows you to specify clock edge activity on another channel. An "and" entry allows you to specify a high level or low level to qualify your clock edges. You can only specify one activity per channel. Once a channel has been used in a specification, it will disappear from the softkey label line.

Press the (RETURN) key; the analyzer will accept your clock, as specified, and show symbols for your specification below the appropriate channel numbers.

SETTING THRESHOLD VOLTAGES

You can specify the threshold voltages that identify logic levels in each of the probe pods. The analyzer offers you the standard ttl and ecl levels, and also allows you to enter any switching threshold of your choice from -10.0V to +10.0V in 0.1V increments. To specify a new threshold level, proceed as follows:

Press the [threshold] softkey.

If you want all probe pods to use the same threshold voltage, press the [all_pods] softkey; otherwise, press the softkey that names the pod you are going to specify.

If you selected the [clock_pod] softkey, you need to identify the probe lines you are going to specify: the lower four channels ([chan 3-0]), the upper four channels ([chan 7-4]), or all eight ([all_chan]).

Now press the [ttl] or [ecl] softkey if you are going to use one of these standard threshold levels. If neither of these keys identify the threshold you are going to use, type in the value of the voltage where you want switching to occur (any voltage between -10.0V and +10.0V, in 0.1V increments).

Now press the [RETURN] key. Your new probe pod threshold will be shown in the display above the channel numbers (bits) affected.

LABEL DISPLAY

The data label display gives you a complete rundown of all of the parameters assigned to any of your labels. You can also test the [activity] of all monitored bits in this display the same as in the [data_pods] display. The following paragraphs describe how to call up a label display for any of your labels, and how to interpret the label information.

The descriptions in the following paragraphs are referenced to the label display example shown in Figure 6-2.

HOW TO CALL UP A LABEL DISPLAY

Press the (display) softkey and then the (label) softkey. You will see each of your labels present on softkeys.

Press the softkey which has the name of the label you want displayed, and then press (RETURN) .

You should see a data label display on screen. The lines of the display provide the following information:

1. The top line block will identify the name of the label, and the bits which were assigned to it. The bits will be identified using "P" or "N". A "P" indicates the bit is interpreted with positive logic. An "N" indicates negative logic.

2. The "Location" line identifies the makeup of the label by pod number and bit number.

3. The "Width" line indicates how many bits are included in the label.

4. The "Logic polarity" line indicates the way that the analyzer will interpret high and low states on each of the bits. If "positive", the labeled probes will be interpreted with positive logic. If "negative", the label will be interpreted with negative logic.

5. The words "ranging label" will appear on the last line of the information block, if this label is entirely within the ranging pod (20-bit pod capable of overview measurements) and starts with bit 0. With such a notation, this label can be used in overview measurements, and in ranging applications in trigger, store, and count specifications.

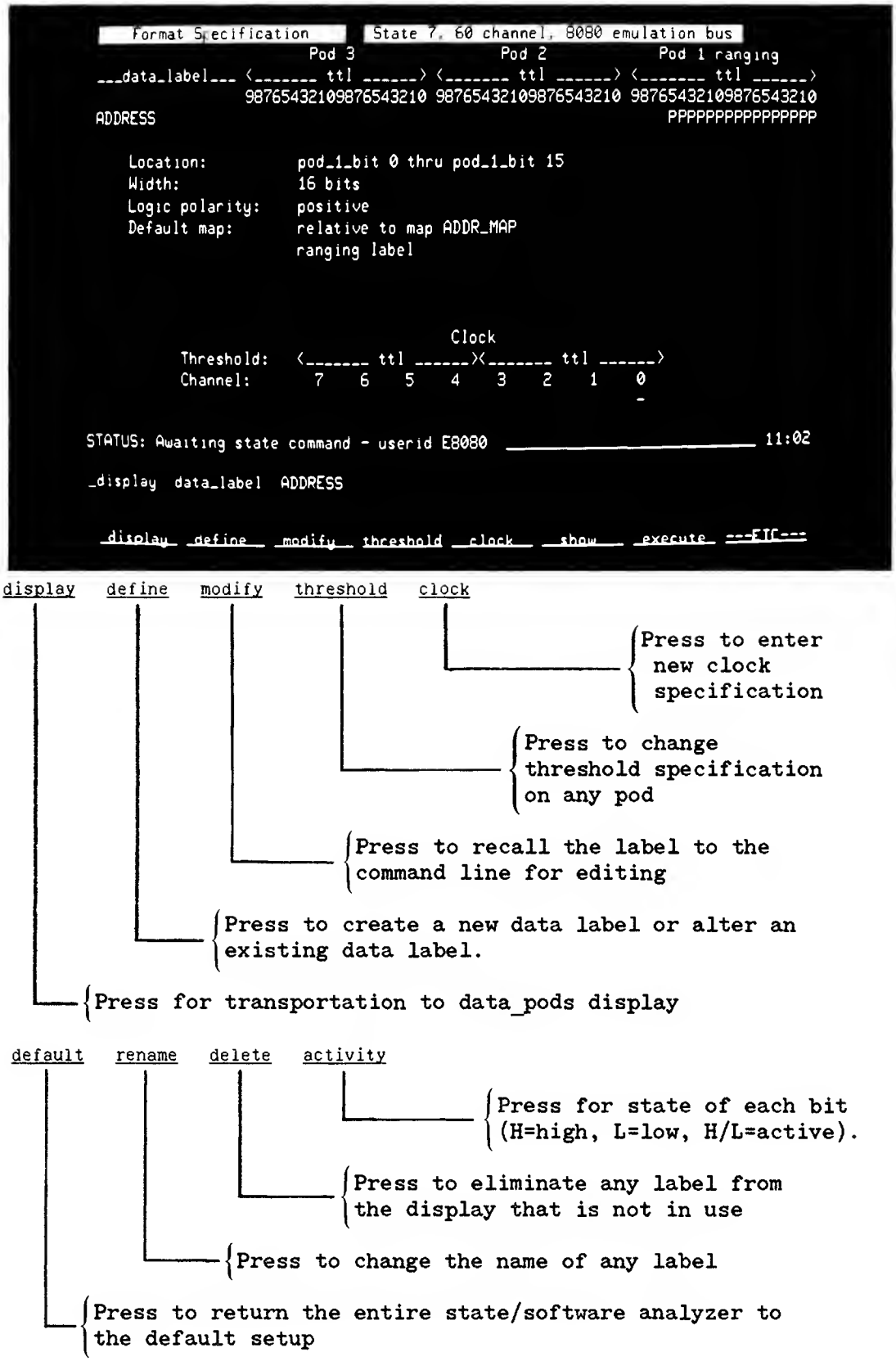


Figure 6-2. The Data Label Display

Chapter 7
THE MAP SPECIFICATION

INTRODUCTION

The map specification is used to create maps of user-defined symbols for expressing states found on labeled sets of bits. To enter the Map Specification, press the (show) and (map_spec) softkeys, and press (RETURN).

A map is a method used to divide an overall range into subranges and points, and to define symbols that represent those subranges and points. An example map for the overall range of address space might include a move routine located in the range of addresses from 1000H to 1040H, stack space in the range of addresses from 7050H to 70FFH, and an unconditional jump located at address 49FFH. These could be entered into a map using the following commands:

(define) MOVE (range) 1000H (thru) 1040H (relative) (start_rng)

(define) STACK (range) 7050H (thru) 70FFH (relative) (end_range)

(define) UNC_JUMP (value) 49FFH

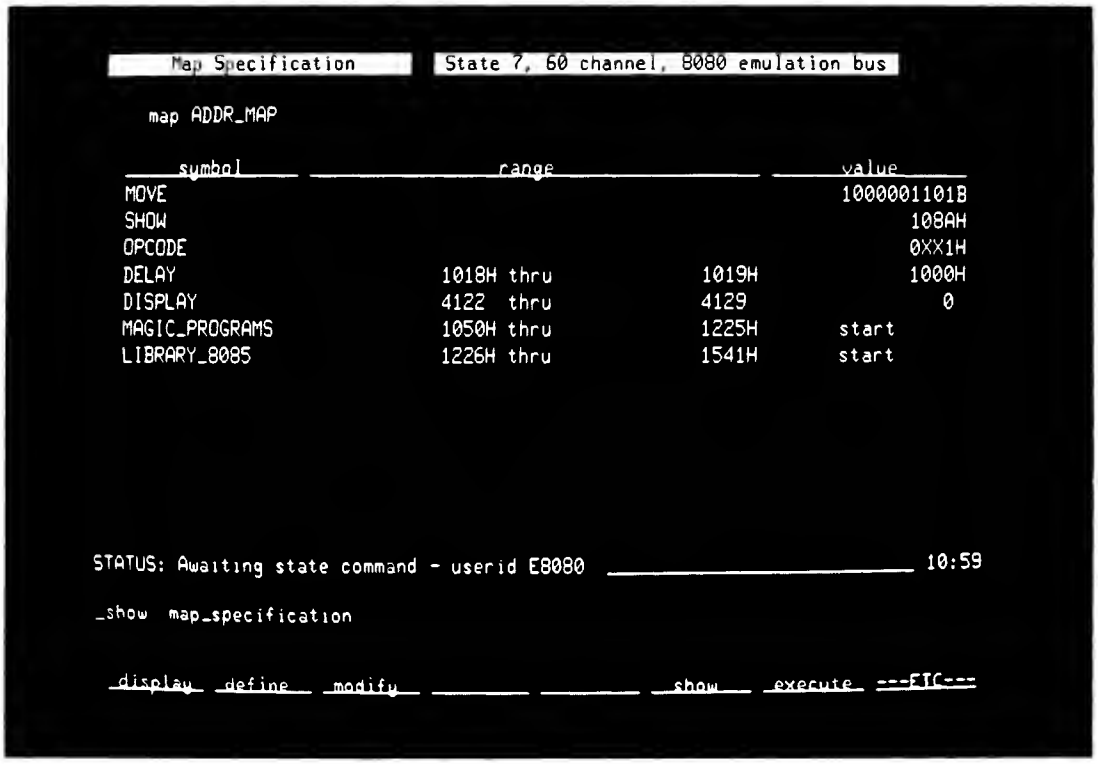
Your map would show:

symbol	range	value
UNC_JUMP		49FFH
MOVE	1000H thru 1040H	start
STACK	7050H thru 70FFH	end

The next step might be to go to the Format Specification and define this map as the default map for your ADDRESS label. Then you could trigger on the twenty-second address in the move routine by specifying: trigger on ADDRESS = MOVE+21 instead of having to calculate the address 1015H in the absolute file. If your analyzer was listing the activity captured during an interrupt execution, it could list STACK-44 in the trace list. This would be more useful than 70D3H in the trace list.

The following paragraphs show you how to create a map, how to enter symbols and assign values or ranges of values in your map, and how to make the analyzer use your map of symbols with a particular label. With this information, you can make your labels and symbols work together when you set up specifications, or when the analyzer formats list displays. Figure 7-1 shows a typical map display.

When you store your instrument configuration in a file by using the (configure) softkey, all of your labels and symbol maps will be stored also. When you reload that configuration into the analyzer, all of your labels and symbol maps will be reloaded just as you had them before.



display define modify

Press to recall a symbol definition to the command line for editing

Press to create a new symbol or alter an existing symbol

Press to show present map in different base or to show a new map

rename delete

Press to eliminate any symbol from the map, or to delete the map

Press to change the name of any symbol or name of the map

Figure 7-1. The Map Display

SYMBOL

The symbol is the name you assign to represent a value or range of values which might be found on a set of labeled probe lines. You might define a symbol like STACK to identify the range of address values in your target system which are allocated to the stack function. This symbol name can be available as a softkey for you to use in your specifications without having to look up the actual address range of the stack. The analyzer will use the symbol in trace lists when it finds corresponding values on the probe lines.

MAP

Each map is a collection of related symbols identified with a map name. The analyzer can store and process a large number of different symbols (typically greater than 100). The symbols can be divided among any number of maps. In the definition of each symbol, you assign a corresponding value, pattern, or range of values. The analyzer will refer to your symbol map both when you are setting up specifications, and when it is formatting a tracelist display. This will appear as follows:

a. If you are setting up a specification, the softkeys will offer you all of the symbols defined in the default map for your label (or defined in any other map of your choice) so you can use them to represent values. For example, you might be using a label in a trigger specification, such as:

   (trigger) (_on_) (ADDRESS) (_=_)

   Your softkeys will show each of the symbols from the default map you assigned to the address label. You might press the UNC_JUMP softkey (if UNC_JUMP is defined in your map): trigger on ADDRESS = UNC_JUMP. The state/software analyzer would then trigger when it found the binary value you defined as 'UNC_JUMP' on the bits you labeled ADDRESS.

b. If the analyzer is formatting a list display, it can show the content of memory using symbols from a map in place of numbers, if available. Figure 7-2 is an example showing the way that the analyzer will use your maps to present a list display. (This figure assumes ADDR_MAP was assigned as the default map for ADDRESS, or you entered the following command:

   (display) (ADDRESS) (relative_to) (ADDR_MAP)

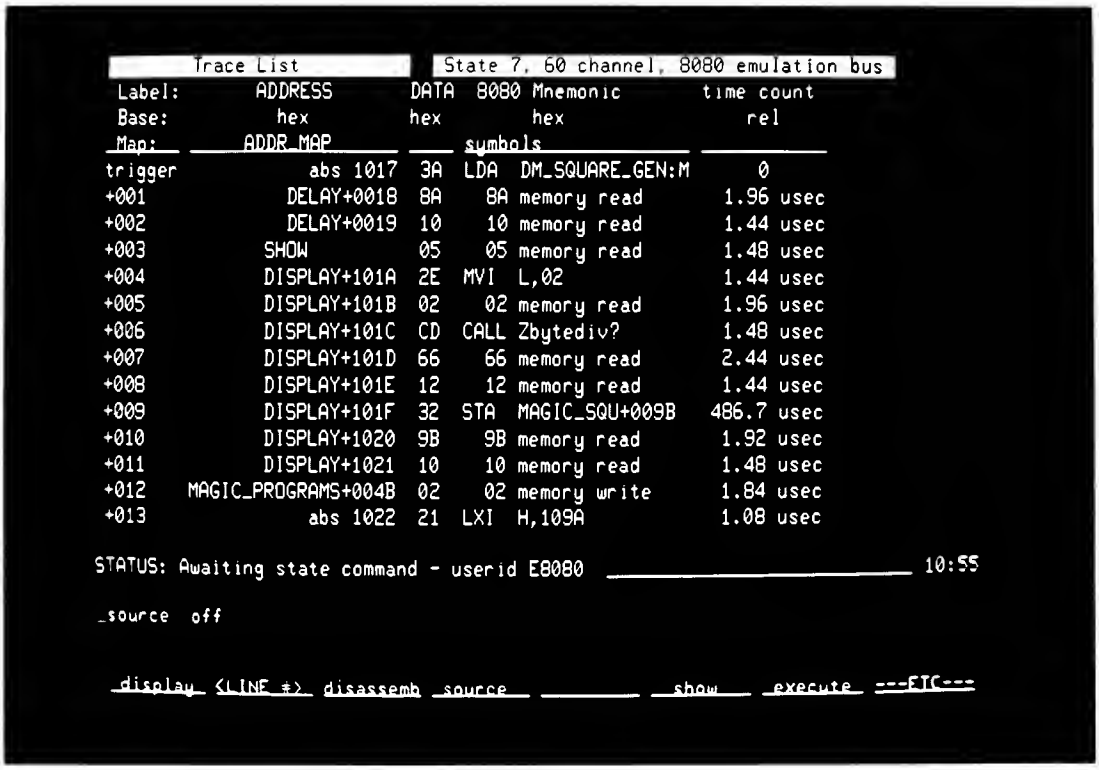


Figure 7-2. Example Display Using Map Symbols

Figure 7-2 shows the ADDRESS and DATA labels which stand for sets of probe leads. Under the ADDRESS label are both numbers and symbols. The symbols are defined on the ADDR_MAP. No map was assigned as the default map for the DATA label so only absolute values are shown.

Under the ADDRESS label, you see the ways the analyzer can use map symbols to represent activity on a label. These are described below, with highest priority first:

1. The symbol name SHOW is the name of a single value on the address map. The symbol SHOW might represent a single value outside of all of the specified ranges, or it might be a value inside of one of the ranges. Note that the name of a single value will override the name of a range of values.

2. The symbol SHOW might also have been defined as a pattern, a value containing one or more X (don't care) bits.

3. A symbol name followed by a number (DELAY and DISPLAY) is the name assigned to a range. The number included with the name tells how far the measured value was offset from the defined reference point of the range. You may have specified that displays of activity in the range shall be counted relative to the start of the range.

The symbol DISPLAY was set up that way. Trace line +006 shows that the address detected is part of the DISPLAY range, and it was 14 higher than the start of the range. The next address was also a member of the DISPLAY range. It was 15 higher than the start of the range.

4. The absolute numbers (abs) listed under the ADDRESS label indicate there were no symbols defined on the map to identify these values.

5. When you select values displayed in ASCII codes, the state/software analyzer will interpret the first seven bits as an ASCII equivalent, and the eighth bit (parity) will be ignored. The entire ASCII character set that is supported by the state/software analyzer is listed in Appendix C at the end of this manual.

Figure 7-2 also shows a column titled 8085 Mnemonic. The entries under this column are obtained from the resident disassembler. Where applicable, these values can also be shown relative to a symbol map.

HOW TO CREATE A MAP

Press the (display) and (map) softkeys and type in the name of the map you are going to create (use from 1 to 16 alphanumeric characters, beginning with an alpha character). Then press (RETURN). The blank map form will be on the display and it will be titled with the name you assigned. The map form consists of the following three columns: symbol, range, and value.

HOW TO ENTER SYMBOLS IN A MAP

You can define symbols on your map using direct keyboard entry. You can also have the analyzer upload symbols directly from the link_symbols table of the absolute file under test.

KEYBOARD ENTRY OF SYMBOLS

Press the (define) softkey and type in the name of the first symbol you want to enter in the map.

Press the (range) softkey if your symbol will represent a range of values, or the (value) softkey if your symbol will represent a single value or pattern. Type in the value, pattern, or range (value thru value).

You can specify symbol values using any of the five number bases available (decimal is the default base), or you can use symbols already defined on a map and the state/software analyzer will use the values defined for those symbols.

The default display of a map shows all symbol values in the bases used when they were defined. If desired, you can have all values expressed in binary, octal, decimal, hexadecimal, or ASCII. If a value cannot be converted to the base selected, dollar signs ($) will be shown in its place.

You can specify a symbol that represents a pattern containing one or more X (don't care) bits. Pattern symbols are useful for triggering on states with certain flag bits set. When the state/software analyzer is preparing a tracelist using a map with two or more patterns that can satisfy a single value, the analyzer will use the first pattern symbol that matches the value.

If you specify a range of values, each point within the range will be identified relative to the range reference point. The range reference point determines how offsets in the range are counted. The default location of a reference point is the start of the range (lo_read+0004 is four values above the start of the range). If you decide to have the count relative to the end of the range, press (relative) and (end_range) when defining the symbol.

The range reference point will affect the way the analyzer interprets your symbol in a specification. If you enter (trigger) (_on_) (ADDRESS) (_=_) (lo_read), the analyzer will trigger on the reference point you defined for the lo_read range.

You can have values within a range identified relative to any value inside or outside the range. This is useful when your symbol represents the range of an array and you want to identify array locations.

Your definition of the range reference point can be a symbol already defined on this or any other map, if desired. The analyzer will use the value of that symbol when computing offsets within your symbol range.

Press the (RETURN) key to enter your symbol in the map. The symbol and corresponding range, pattern, or value you specified will be shown on the map. The numbers appearing in the range or value columns will be in the number base you used in your definitions. For ranges, the value column will identify the range reference points.

Repeat the foregoing procedure to enter as many new symbols as required in your map. The symbols will be sorted by range and value, except for the symbols containing X's (don't cares).

UPLOADING SYMBOLS FROM ABSOLUTE CODE FILES

The following line is an example of a command you might use to enter a symbol into a map from the link_symbols table of an absolute file. MONITOR is a range of values. The range begins with the value of the address where the MONITOR segment begins in the absolute code and ends where the MONITOR segment ends in the absolute code.

define MONITOR range file MONITOR:TRACE start thru file end

The word "start" was unnecessary in the above line because start is assumed in this kind of definition, unless otherwise stated.

Because no different file name was used to define the end of the MONITOR range, the state/software analyzer assumes that the most recent file name (MONITOR:TRACE) is the one intended.

define PARSE_STATE value PARSE_STATE global

With the above command, the state/software analyzer will access the link_symbols table for the absolute file and find the address value of the PARSE_STATE global symbol. It will enter the symbol "PARSE_STATE" into your map along with the address it found in the link symbols table.

Symbols that identify ranges in absolute files begin with your symbol name and end with compiler-assigned 'R' and 'E' symbols. For example, you might have defined the symbol MOVE to identify a function in the absolute file. The compiler will assign RMOVE to identify the first byte of the ending instruction in the MOVE routine, and EMOVE to identify the last byte of the ending instruction in the MOVE routine. Using these symbols, the analyzer can find the addresses in the absolute file that contain the bytes which begin and end this routine when you upload the MOVE symbol into a map in the state/software analyzer.

Be sure to upload the 'R' and 'E' symbols associated with any range symbols you upload into a map. The reason that the compiler assigns both the 'R' and 'E' symbols to bytes in the ending statement is to accommodate those systems that do not place the address of every byte on the address bus. In those systems, the first byte of the ending statement for the MOVE routine (RMOVE) will be addressed, but the last byte may not be. The compiler assignments allow the use of symbols for ending statement recognition, whether the last byte is addressed on the address bus or not.

WHY UPLOAD SYMBOLS FROM ABSOLUTE CODE

There are at least two good reasons for uploading symbols from absolute files into maps of the state/software analyzer. The reasons are as follows:

1. If your state/software analyzer does not have a memory expander, it cannot access the link_symbols table for the absolute file to obtain symbols when it is formatting a trace list. If you upload these symbols into a map, the state/software analyzer can use the map when presenting the trace list.

2. If there are symbols in the absolute file that you are using in analyzer specifications, you might like these symbols to appear on softkeys. If you upload these symbols into a map, they will appear on the softkeys.

There is at least one good reason *not* to upload symbols from your absolute code into a map. When you make changes in your software and relink the code, the absolute addresses of the symbols will probably change. These changes will be recorded in the new link_symbols table, but not in your map. After the linking is complete, you will have to recheck each symbol and upload again.

RESTRICTIONS TO MAP ENTRIES

There are very few restrictions to observe when you set up a symbol map. You cannot assign symbols with overlapping ranges (e.g. you cannot have a symbol called Y with a range of 4000H to 4FFFH, and another symbol called Z with a range of 45FFH to 55FFH.) This is because the analyzer will not know which symbol to place on a tracelist when it finds a number such as 4800H.

A map cannot have two symbols with the same values. For example, you cannot define RUN to be value 0XXH and JSM to also be 0XXH on the same map, but you can define a symbol for 0XXH and another for 01XH on the same map because these two values are not identical.

You cannot use the same name to identify both a map and a symbol in a map. This is because the analyzer will not know whether you are calling up a map or a symbol when it reads the name.

Each map is a separate unit. There are no restrictions about avoiding values, patterns, ranges, or symbols that appear on other maps.

HOW THE ANALYZER USES YOUR MAP

When the state/software analyzer uses your map, it will make up to three passes for each symbol or value. On the first pass, it will check only single-value symbols. If it finds a match, it will accept the associated symbol or value. If not, it will begin pass two.

On pass two, it will check symbols that represent patterns containing don't care bits. It will accept the first symbol it finds that matches the present value. All other patterns will be ignored, even if they also match the present value. If the analyzer does not find a match, it will begin pass three.

On pass three, the state/software analyzer will check the symbols for ranges. If it finds a match, it will use the associated symbol or value and offset from the reference point of the range. If it does not find a match, it will exit the map search routine. Flow charts detailing these procedures are shown in Chapter 8A and Chapter 10; these show how the analyzer converts symbols to values in specifications and values to symbols in tracelists, respectively.

Chapter 8
THE TRACE SPECIFICATION

The trace specification is where you define the type of measurement to be made and the parameters to be included in the measurement. Because of the measurement versatility available in the trace specification, this chapter has been divided into six sections (Chapters 8A through 8F). Each section discusses a different aspect of the measurement or control features available in the trace specification.

The trace specification is the specification placed on screen when you first enter the state/software analyzer. It can also be brought to the screen by the command:

(show) (tracespec) (RETURN)

The default form of the trace specification presents the Trigger, Store, and Count functions, which are described in Section 8A. The Sequence, Overview, Assert, and Master functions are described in other sections of Chapter 8. These are not active in the defaulted form of the trace specification. To activate any of these functions, you can enter commands in the trace specification which call for their use.

Figure 8-1 shows the trace specification set up to make a typical trace measurement. The measurement will be triggered when the state/software analyzer finds a value equal to the symbol MON_ENTRY on the ADDRESS probe lines. The trigger will be stored as the first state in the trace memory. The remainder of the trace memory will be filled with states whose address values are equal to the symbols FLOAT_ADD and STACK.

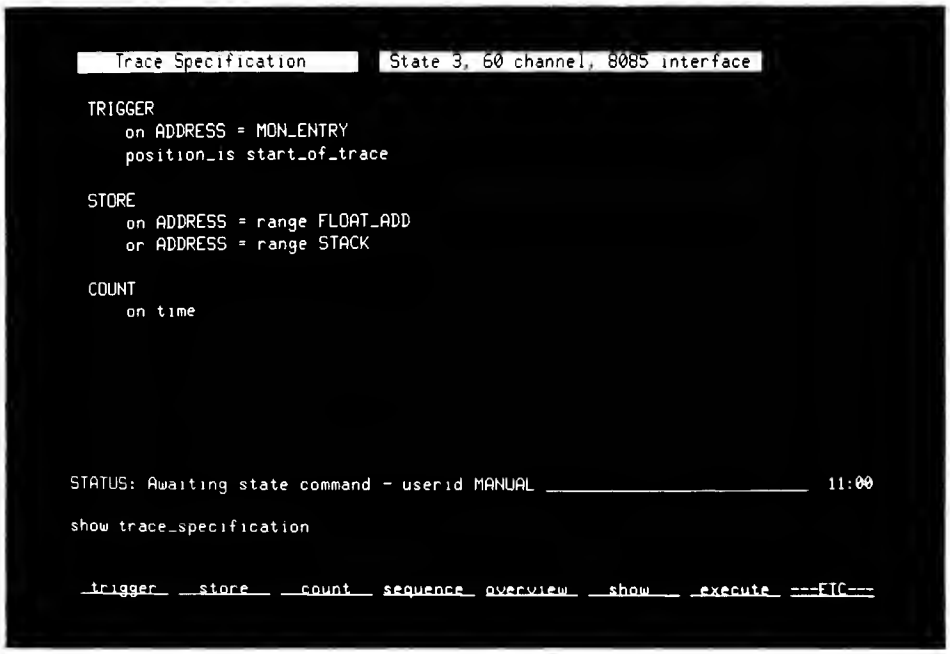


Figure 8-1. Trace Specification

Chapter 8A
TRACE MEASUREMENTS

INTRODUCTION

A trace is a measurement that collects a series of states. Each state describes the conditions at several points in a system under test. The series of states shows how conditions changed with time at each of the monitored points.

Trace measurements are made through a set of probe channels. Each probe channel is connected to monitor activity at a point in a system under test. When a trace measurement is complete, the trace memory will contain a set of states captured from the probed points. These states are displayed in a trace list.

Example specification:

(trigger) (_on_) (ADDRESS) (_=_) 1000H

(trigger) (position) (center)

(store) (_on_) (ADDRESS) (_=_) (range) 0F000H (thru)

0FFEFH (and) (STATUS) (_=_) (Write)

(count) (_on_) (time)

With the above specification, the state/software analyzer will begin capturing states into memory as soon as you press the (execute) softkey. It will only store those states that have addresses from 0F000H through 0FFEFH and are write transactions. The state/software analyzer will measure the time between each of the states that it stores in memory. When it finds 1000H on the address lines, it will capture that state into memory and identify it as the trigger. Then it will continue to capture an additional number of states equal to half the size of the trace memory. This leaves the trigger state in the center of memory.

This chapter describes the measurement parameters available for collecting states into the trace memory. The three parameters are trigger, store, and count. Each of these parameters is discussed in detail in this section. Several examples of how to use these parameters to make trace measurements are given at the end of this section.

TRACE MEASUREMENT FUNCTION

Trace measurements in the state/software analyzer are made with three basic parameters and a 256-deep state memory. The basic parameters are trigger, store, and count. They select the information that is stored in the trace memory. Figure 8A-1 is a diagram showing the elements involved in the trace measurement function of the state/software analyzer.

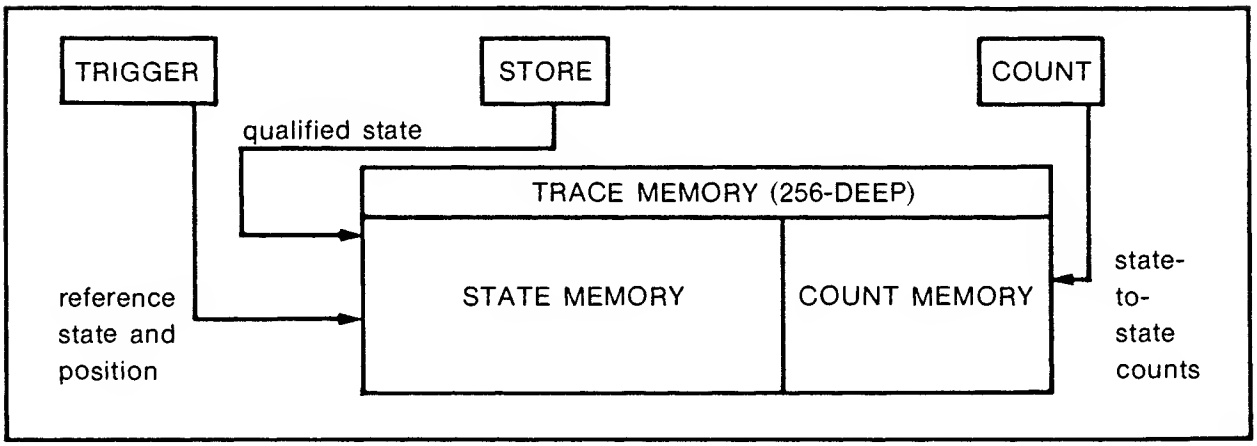


Figure 8A-1. Trace Measurement Functions

When a trace starts, the state/software analyzer examines each state that it obtains from the probes. It stores into the trace memory the trigger state plus every state that meets the store specification. The default specification for the store function qualifies all states to be stored in trace memory. You can enter a special store specification if you want to store only certain states in the trace memory.

The trigger function also examines each state from the probes. When it finds the first state that meets the trigger specification, it stores that state in memory. The default specification for trigger is any state meets specification. You can enter special conditions to be met by the trigger state if you want to examine a specific area of program activity. The trigger state will always be stored in trace memory even if it does not meet the store specification.

Now the state/software analyzer will fill the remainder of memory with states that meet the store specification. The trigger position specification will determine how many more states will be stored. If you selected the "start of trace" as your trigger position, 255 more qualified states will be stored. If you selected "end of trace" as your trigger position, the trace will complete immediately. The default of trigger position is "start of trace".

The count function places a time count or state count in the trace memory with each stored state that is count-qualified. The count shows either elapsed time since the last state was counted, or identifies count-qualified states and/or shows how many have been found since the trace began. The default of the count function causes it to measure the time between each state stored in the trace memory.

TRACE SPECIFICATION TERMS

The following terms appear when you are entering parameters for the trigger, store, and count specifications. You should understand these terms before trying to set up unique parameters for the trigger, store, and count specifications.

1. on - prepares state/software analyzer to accept specification of a state or range of states on which to perform the associated function.

2. modify - Calls the present specification for the associated parameter to the command line so you can make changes to it, if desired.

3. position - prepares state/software analyzer to accept specification of where in the trace memory you want the trigger state to be located.

4. any_state - allows any state to satisfy the corresponding specification.

5. nothing - prevents trigger recognition. Use this entry to make the state/software analyzer store states leading up to loss of the system clock.

6. ADDRESS (or)
   POD1 (or)
   <LABEL> - labels that identify sets of probe channels.

7. "=" - Equal sign, used to assign a value or range to be found on a labeled set of probe lines, such as "trigger on ADDRESS = 1040H".

8. "<>" - Not-equal sign, used to identify values or ranges to be ignored on a labeled set of probe lines.
   Example: store on ADDRESS <> 5400H.
   The trace memory will store all activity except that having a hexadecimal address of 5400H.

9. any_value - allows any value to satisfy the corresponding specification.

10. range - a series of values with no breaks in the series and no X (don't care) values on any of the specified probe lines.

11. VALUE - prompts you to enter a number or a character string to identify a combination of 1 and 0 states on a labeled set of probe lines.

12. PATTERN - prompts you to enter any value as defined above, or any combination of 1, 0, and X (don't care) states.

13. SYMBOL - prompts you to enter the name of any map symbol or file-based symbol used to identify a value or range of values. Refer to the Map Specification Chapter.

14. global - designates a symbol as global to all source files linked within the absolute file under test. The symbol '?' has the same meaning as 'global' in a command.

15. map - allows you to specify a map other than the default map as the source of your symbol. Refer to the Map Specification Chapter.

16. line - used to specify a line of code from a compiled source file. (Example: line 4 end specifies the last addressable byte or word emitted by the fourth line of the high level source file.) The symbol '#' has the same meaning as 'line' in a command.

17. file - used to specify the name of a source file within the linked absolute code under test. When used with a file name, it selects a new file. When used without a name, it identifies the most recently named file. It can be used to override the default map for a label, if desired. (Example: MOVE file MONITOR specifies the MONITOR file as the location of the MOVE symbol). The symbol ':' has the same meaning as 'file' in a command.

18. FILE - prompts you to enter the name of a source file.

19. start - file 'start' indicates the first addressable byte or word emitted at the specified point in the source file. map-symbol 'start' indicates the first addressable byte or word in the range of the map symbol.

20. end - file 'end' indicates the last addressable byte or word emitted at the specified point in the source file. map-symbol 'end' indicates the last addressable byte or word in the range of the map symbol.

21. prog - specifies use of the program segment of the absolute (object) code.

22. data - specifies use of the data segment of the absolute code.

23. comn - specifies use of the common segment of the absolute code.

24. EXPRESSION- Arithmetic or logical operator. The state analyzer will accept +, -, *, /, (,), mod, &, and |. These are defined as follows:

+	arithmetic addition
-	arithmetic subtraction
*	arithmetic multiplication
/	arithmetic integer division
(	left parenthesis for grouping
)	right parenthesis for grouping
mod	arithmetic division remainder
&	logical AND
|	logical OR

COMBINING TRACE SPECIFICATION TERMS

You can group entries for trace specification terms by using the "and" and "or" softkeys. These softkeys appear at appropriate points during specification entry. They provide the functions described below:

and - The "and" entry combines two or more activities into a single specification, as follows:

```
(trigger) (  on  ) (ADDRESS) (  =  ) 1000H  (and)

          (STATUS) (  =  )  4H
```

The above line requires trigger recognition only if 1000H is on the ADDRESS lines within the same state that 4H is on the STATUS lines.

You can include all of your labels in an "and" entry as long as none of the labels overlap. For example, if all of the status lines were labeled STATUS, and the read/write line was labeled R_W, you could use either STATUS or R_W in an "and" specification, but not both.

or - The "or" entry is used to enter two or more states, either one of which can satisfy the specification, as follows:

(trigger) (on) (ADDRESS) (==) 1000H (or)

(ADDRESS) (==) 1010H

The state/software analyzer will recognize 1000H or 1010H as the trigger state if either one is found on the ADDRESS lines.

You can combine "or" and "and" entries as follows:

(trigger) (on) (ADDRESS) (==) 1000H (and) (STATUS) (==)

(R/W) (or) (ADDRESS) (==) 2000H

(and) (STATUS) (==) (Opcode)

STATE-RECOGNITION RESOURCES

There are eight state-recognition resources available in the state/software analyzer. All eight can recognize state patterns on the incoming probe lines. Four can also recognize ranges of states on ranging labeled probe lines. These resources are used to qualify the trigger, store, and count parameters during trace measurements. The following breakdowns list the capabilities of these resources:

1. Pattern-Recognition:

   a. Eight separate resources.

   b. All can recognize values which match single patterns (composed of 1, 0, and X, binary).

   c. All can recognize "=" values.

2. Range-Recognition:

   a. Four of the eight resources can also recognize ranges of values.

   b. These four can recognize any value as long as it falls within the specified range of values.

   c. All can recognize "=" and "<>" ranges of values.

d. The range must be specified to be found on a ranging label obtained from a set of probe lines within a ranging pod that is installed as pod 1.

NOTE

The 20-channel Acquisition Board must be installed in order to obtain the Ranging and Overview functions. No range recognition is available when the Overview function is turned on.

3. Not-Equal "<>" Pattern Recognition:

   a. By combining one of the four range-recognition resources with one of the other four resources, the state/software analyzer can compose a resource capable of recognizing "<>" patterns on any set of incoming probe lines.

   b. Only four resources are available when all are used to recognize "<>" patterns.

The eight state-recognition resources are shared among the trigger, store, and count specifications. You can use all eight to make up the trigger specification; the store and count functions will still run at their default specifications. You can use all eight to make a complex store specification; the trigger and count functions will run at their default specifications. Only four can be used by the count specification. This leaves four available for sharing among the trigger and store specifications. Figure 8A-2 is a diagram showing how the state-recognition resources can be shared among the trigger, store, and count specifications.

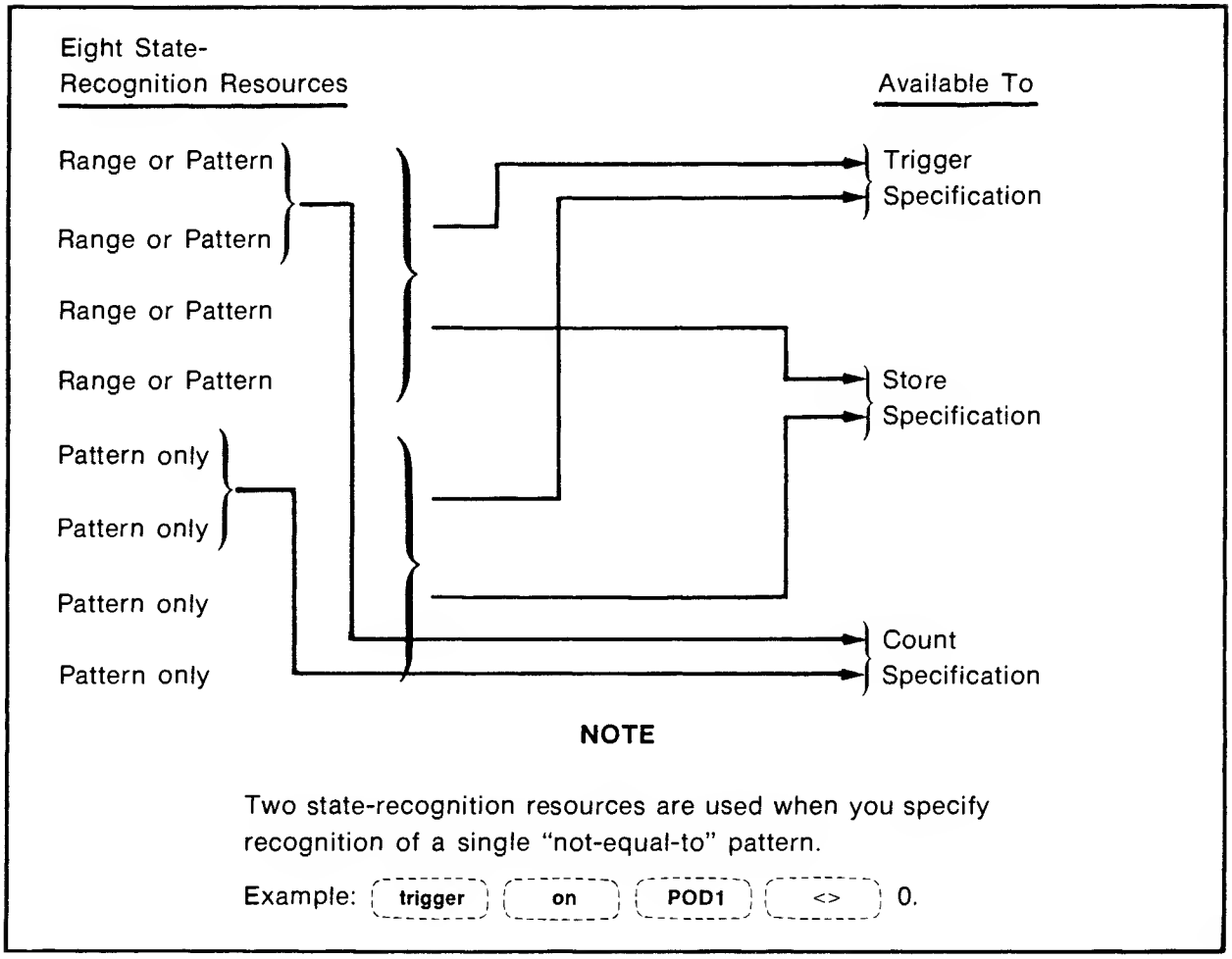


Figure 8A-2. Allocating State-Recognition Resources

USING STATE-RECOGNITION RESOURCES

The following examples show how the state-recognition resources can be used in trace specifications:

Example 1: Using All Range-Recognition Pattern-Recognition Resources

```
TRIGGER
   on ADDRESS = 55XXH

STORE
   on ADDRESS = range 1000H thru 1040H
   or ADDRESS = range 2000H thru 2040H
   or ADDRESS = range 3000H thru 3040H
   or ADDRESS = range 4000H thru 4040H
   or ADDRESS = 55XXH
   or ADDRESS = 66XXH
   or ADDRESS = 77XXH

COUNT
   on time
```

In example 1, four range-recognition resources are used for store. Four pattern-recognition resources are also used (one for trigger, and three for store). This example shows ORing store terms to capture different areas of address space.

Example 2: Using State-Recognition Resources That Include One <> Range

```
TRIGGER
   on ADDRESS = range 1000H thru 2000H
   or ADDRESS = range 3000H thru 4000H
   or ADDRESS = range 5000H thru 6000H
   or ADDRESS = 2050H
   or ADDRESS = 3050H

STORE
   on ADDRESS <> range 6001H thru 0FFFFH
   or ADDRESS = 0XXXH
   or ADDRESS = 1XXXH

COUNT
   on any_state
```

In example 2, four range-recognition resources are used (three for trigger, and one for store). Four pattern-recognition resources are also used (two for trigger and two for store). One of the range-recognition resources was used for "<>" range recognition. (Range recognition resources can recognize either "=" or "<>" ranges.) No state-recognition resources were available for use by the count specification.

Example 3: Using Range-Recognition Resources For Single Address Recognition

```
TRIGGER
   on ADDRESS = range 1000H thru 2000H
   or ADDRESS = 20X0H
   or ADDRESS = 30X0H
   or ADDRESS = 40X0H
   or ADDRESS = 50X0H
   or ADDRESS = 60X0H

STORE
   on any_state

COUNT
   on ADDRESS <> 1XXXH
```

In example 3, one range-recognition resource, and five pattern-recognition resources were used in the trigger specification. One range-recognition resource was combined with one pattern-recognition resource to make up the count specification (it contains a "<>" pattern). The store specification does not use any of the state-recognition resources.

TRIGGER

Normally, a trigger must be found in order to complete a trace. The trigger defines a point of reference within trace memory. There are two aspects of triggering that you can control when you specify a trigger: what position that trigger should occupy in trace memory, and what activity you want the analyzer to recognize as its trigger.

SELECTING TRIGGER POSITION

You can specify any position in the trace memory for your trigger. This allows you to gather a desired number of pretrigger and posttrigger states for analysis. The following paragraphs describe specifications you can use to select trigger position. See figure 8A-3.

(trigger) (position) (start). This is the default specification for trigger position. If you select start of trace as your trigger position, the trigger will be the first state stored in memory. The remainder of memory will be filled with store-qualified states that occur after the trigger. (Trigger state is line 0. Remainder of memory is filled with states +1 through +255.)

(trigger) (position) (end). If you select end of trace as your trigger position, the trigger will be the last state stored in memory. The remainder of memory will be filled with store-qualified states that occurred before the trigger. (Trigger is line 0. Remainder of memory is filled with lines -255 to -1.)

(trigger) (position) (center). If you select center of trace as your trigger position, the trigger will be the center state in memory. Half of the memory will contain store-qualified states that led up to the trigger, and the other half of the memory will contain states that occurred after the trigger. (Trigger is line 0. Remainder of memory is filled with lines -128 through +127.)

(trigger) (position) <NUMBER> (after/before). You can position your trigger state anywhere within the memory so that you can gather a particular number of pretrigger or posttrigger states by use of the (after), and (before), softkeys. ((trigger) (position) 44 (after) places your trigger on line 0. The remainder of memory is filled with lines -43 through +212.)

NOTE

Specifying "trigger position_is 1 state_after_start" is the same as specifying "trigger position_is start_of_trace".

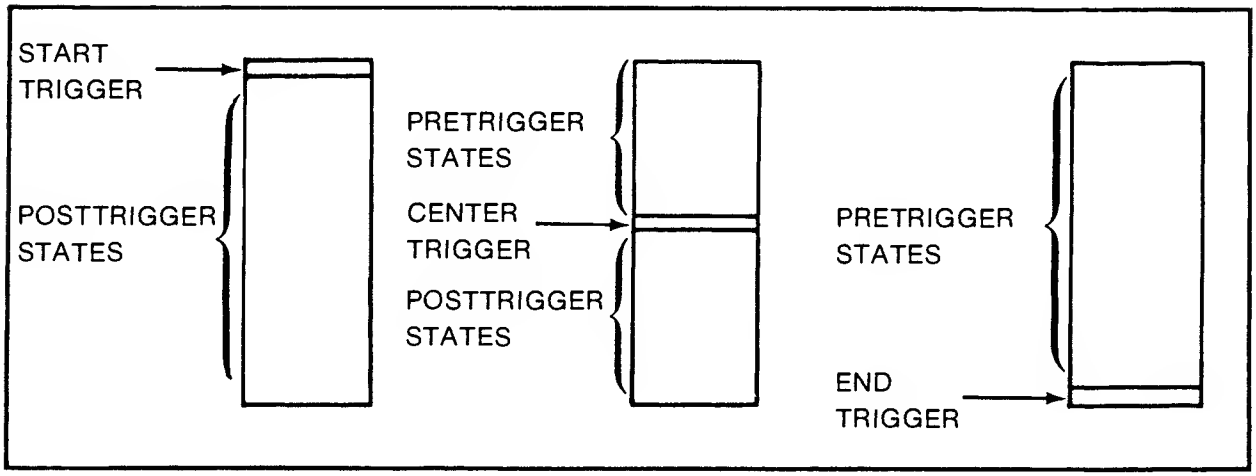


Figure 8A-3. Selecting Trigger Position

SELECTING TRIGGER POINT

The purpose of this paragraph is to show you how to set up the state/software analyzer to trigger a trace when selected software is executed. The basic trigger is a software occurrence which is found on a set of input probe lines. You can specify the probe lines to be monitored for the trigger occurrence, and you can specify the software state to be recognized as the trigger. You can select a single state to be recognized as the trigger, or you can set up the state/software analyzer to recognize its trigger the first time it finds any state within a range of states or within a pattern of states.

You can have the analyzer trigger a trace when the sequence or one of the window elements recognizes a specification ((trigger) (_on_) (window) (one) (disable)). Refer to the sequence and window section of this chapter for details on windowing.

You can have trigger search enabled in the state/software analyzer by the sequence or one of the window elements. Refer to the sequence and window section of this chapter for details.

You can have the analyzer trigger a trace when one of the overview events occurs during an overview measurement ((trigger) (_on_) (overview) 1). Refer to the overview measurements section of this chapter for details.

You can have the analyzer "trigger on nothing". This specification keeps the analyzer from finding a trigger. With this specification, the analyzer will gather a continuous stream of states until you press the (halt) key or the target system clock stops. This allows you to gather 256 states leading up to a system failure.

You can use a symbol, such as KEY_INPUT, to represent the value to be recognized as the trigger. The way you specify the symbol will determine which value or values will be recognized as the trigger. Assume that KEY_INPUT is defined in the default map for ADDRESS as the range of addresses from 03FAH to 04A1H, measured relative to the start of the range.

a. (trigger) (_on_) (ADDRESS) (_=_) (KEY_INPUT). The analyzer will trigger when it finds 03FAH on the lines you labeled ADDRESS. This is the first address in the KEY_INPUT range.

b. (trigger) (_on_) (ADDRESS) (_=_) (KEY_INPUT) (end). The analyzer will trigger when it finds 04A1H on the lines you labeled ADDRESS. This is the last address in the KEY_INPUT range.

c. [trigger] [on] [ADDRESS] [==] [range] [KEY_INPUT]. The analyzer will trigger the first time it finds any value between 03FAH and 04A1H on the lines you labeled ADDRESS.

d. [trigger] [on] [ADDRESS] [==] [KEY_INPUT] [+] 15H. The analyzer will trigger on 3FAH + 15H = 40FH.

e. [trigger] [on] [ADDRESS] [==] [DISPLAY]. When DISPLAY is specified in a map as being a single pattern made up of 1, 0, and one or more X (don't care) bits, its use allows trigger recognition on two or more values. You might use an entry like this to trigger during specific types of executions. For example, you can specify the bits that identify an opcode fetch, and assign X to all other bits in the label. Using this symbol, you can trigger on any state that includes an opcode fetch.

You can use symbols from several different sources to represent trigger values.

a. [trigger] [on] [ADDRESS] [==] INTERRUPT. The analyzer will find the value of the INTERRUPT symbol in the default map for ADDRESS. When it finds the same value on the ADDRESS lines, it will trigger.

b. [trigger] [on] [ADDRESS] [==] UPDATE [map] [PROCESS]. The analyzer will find the PROCESS map (not the default map for ADDRESS) and find the value of the UPDATE symbol in that map. The analyzer will trigger when it finds that same value on the ADDRESS lines.

c. [trigger] [on] [ADDRESS] [==] INTERRUPT [global]. The analyzer will find the value of the INTERRUPT symbol designated global in the absolute code under test. It will trigger when it finds that value on the ADDRESS lines.

d. [trigger] [on] [ADDRESS] [==] MOVE [file] DISPLAY. The analyzer will determine the address space occupied by the DISPLAY source file in the absolute code under test. It will find the address of the first byte or word emitted by the MOVE symbol in the DISPLAY source file. It will trigger when it finds that address on the ADDRESS lines.

You can trigger on lines of source code represented in the absolute code under test.

a. [trigger] [_on_] [ADDRESS] [__=__] [file] KEYBOARD [prog] [end] -7. The analyzer will determine the address range in absolute code that is occupied by the program segment of the KEYBOARD source file. It will trigger when it finds the last address in that range on the ADDRESS lines.

b. [trigger] [_on_] [ADDRESS] [__=__] [file] SCRATCH_PAD [data] [start] + 5. This entry will cause the analyzer to determine the value of the first address in the data segment of the SCRATCH_PAD source file within the absolute code under test. It will add five to that value and trigger when it finds the new value on the address label. When specifying trigger occurrence on a line of source code in the absolute code, you must specify whether that line is part of the program segment, the data segment, or the common segment.

c. [trigger] [_on_] [ADDRESS] [__=__] [line] 4 [end]. The analyzer will determine the address range occupied by the object code emitted from the fourth program line of the default file. Trigger will occur on the last addressable byte or word within that address range. Note that the "default file" is the last file specified in any preceding entry. All following entries will be assumed to be part of that file until a new file is named.

You can trigger on the first occurrence found within selected ranges of the absolute code under test.

a. [trigger] [_on_] [ADDRESS] [__=__] [range] [file] KEYBOARD. The analyzer will determine the address space occupied by the KEYBOARD module in the absolute code under test. It will recognize trigger on the first occurrence of any address from that range.

b. [trigger] [_on_] [ADDRESS] [__=__] [range] [file] KEYBOARD [data]. The analyzer will determine the address space occupied by the data segment of the KEYBOARD module in the absolute code under test. It will recognize trigger on the first occurrence of any address in that range.

c. [trigger] [_on_] [ADDRESS] [__=__] [range] [line] 75. The analyzer will determine the address space occupied by the absolute code which was emitted from high-level line 75 in the default file (most recently named file). It will trigger on the first occurrence of any address in that space.

d. [trigger] [_on_] [ADDRESS] [__=__] [range] T [thru] Y. The analyzer will check the default map for the ADDRESS label. If either T or Y (or both) are defined there, it will use the corresponding values in this specification. If either symbol (or both) are not defined in the default map for the ADDRESS label, the analyzer will check the link symbol map for the absolute code file when the

measurement begins. First, it will check through the global symbols to see if T and Y are defined. If they are not there, it will check the local symbols within the default file (last file named in a specification). The analyzer will accept the values of T and Y even if one of these symbols is defined in one source and the other symbol is defined in a different source (map, global, or local).

e. ⟨trigger⟩ ⟨on⟩ ⟨ADDRESS⟩ ⟨=⟩ ⟨range⟩ ⟨file⟩ KEYBOARD ⟨comn⟩ ⟨thru⟩ ⟨file⟩ DISPLAY ⟨comn⟩. The analyzer will determine the address where the common segment of KEYBOARD begins and where the common segment of DISPLAY ends in the absolute code. It will use these two addresses as the boundaries of a range within which any address value will be accepted as trigger.

NOTE

The above command line uses one range resource. The same trigger specification could be entered by specifying ⟨range⟩ KEYBOARD ⟨comn⟩ ⟨and⟩ ⟨range⟩ DISPLAY ⟨comn⟩ (assuming these are contiguous file spaces), but this specification would use two range resources.

You can use compiler-generated symbols in your specifications.

When the HP 64000 compiles absolute object code from a high level source file, it generates special 'R' and 'E' symbols to identify the endings of any procedures, functions, and programs that you identified by symbols. These 'R' and 'E' symbols are assigned to the first and last bytes emitted by the ending instruction. For example: if you have a procedure that you have identified using the symbol MOVE, the compiler will assign RMOVE and EMOVE to the first and last bytes, respectively, of the ending instruction in that procedure.

The EMOVE symbol is useful when you are measuring execution times. You can measure from MOVE to EMOVE and include every clock cycle in the execution of the procedure.

The RMOVE symbol is useful when you are using a compiler for a microprocessor that does not use all numbers in its address scheme. For example, the 68000 address counter does not use odd numbers in its address scheme. To memory location 0, it sends both byte 0 and byte 1. If you specify trigger on the 'E' symbol of some procedure (such as EMOVE), the analyzer may never trigger. In this case, the ending instruction of that procedure is two bytes long so the address of the second byte (identified by the 'E' symbol) will never appear on the bus. In order to trigger on the end of the range of that symbol, you will have to specify the 'R' symbol (such as RMOVE). Its address will appear on the bus of any processor.

STORE

The purpose of this paragraph is to show you how to set up a store specification that makes the state/software analyzer store states of interest in the trace memory. The default store specification makes the state/software analyzer store every state. When the default specification fills the trace memory with unwanted states, you can specify the kinds of states to be stored, and the state/software analyzer will store only those states.

The store specification is made up of two parts: a qualification, and an enable. The qualification is a set of conditions to be met by a state before it is stored in trace memory. The enable is a window in program activity where the analyzer is able to store qualified states, and outside of which, storage is disabled. The default is all states are qualified, and the state/software analyzer is always enabled.

SELECTING STORE QUALIFICATION

The qualifications you use to allow storage of states can be based on certain addresses being executed in the software, or the presence of a certain flag on a status bus, or any of a variety of conditions in the probe channels (store) (on) (ADDRESS) (==) (range) (DISPLAY) (or) (STATUS) (==) (Opcode).

You can also use the sequence and window elements to qualify storage of states (store) (on) (sequence) (enable). Refer to the sequence and window section in this chapter for details.

When you execute a trace, the analyzer will examine each state. If the specified qualifications are met and storage is enabled, the state will be stored in memory. If the qualifications are not met or storage is not enabled, the state will be ignored.

All of the capabilities for using symbols described in the trigger paragraph are also available for specifying store qualifications.

SELECTING STORE ENABLES

Store enables usually involve specifying a window in software where you want the analyzer to store states. You enter the beginning and ending addresses of the window in the sequence (or window) specification. Then you enable storage during the enable period from that sequence or window element. Refer to Chapter 8B for details of how to set up and use the sequence and window elements.

COUNT

The count function can be used to measure periods of time or numbers of transactions between states stored in memory. If you measure time, the time will be measured between each of the states stored in memory. If you count states, you can have each state counted, or you can have the analyzer count only the occurrences of certain kinds of states. You might only want to count the number of times 1000H appears on the probe lines between 2000H and 4000H (count) (on) (POD 1) (=) 1000H, (store) (on) (POD 1) (=) 2000H (or) (POD 1) (=) 4000H). The default count specification is count time.

You can also enable and disable the count function by controlling it from the sequence or one of the window elements. Refer to Chapter 8B for details of how to use sequence and window elements.

All of the capabilities of using symbols described in the trigger paragraph are available for specifying states to be counted.

HOW THE ANALYZER CALCULATES SYMBOL VALUES

Before trace start, the analyzer executes the routine shown in Figure 8A-5 to determine the binary states represented by all symbols entered in specifications. This routine is performed each time you press the (RETURN) key to enter a specification.

After trace start, the analyzer executes the routine shown in Figure 8A-6 to determine the binary states of any symbols not found before trace start. This routine is performed when you press the (EXECUTE) and (RETURN) keys.

EXAMPLES

TRIGGERING ON AN ADDRESS

You might want the state/software analyzer to trigger a trace when 04FFH appears on the address bus. If you have created a label to identify the lines of the address bus, such as ADDRESS (refer to Chapter 6), you can use that label in your command line, as follows:

(trigger) (_on_) (ADDRESS) (__=__) 04FFH

If you created a symbol map to support activity found on the address bus, and if value 04FF is defined as the start of the DELAY routine in the system under test, then you can use the label and symbol together in your command line, as follows:

(trigger) (_on_) (ADDRESS) (__=__) (DELAY)

In the above command lines, the state/software analyzer will use Format information to determine which probe lines are included under the ADDRESS label, and monitor only those lines for a trigger. It will also determine the binary value which is represented by 04FFH or the DELAY symbol, and recognize trigger when it finds that value on the appropriate probe lines. You can make a trace measurement with the trigger described above, as follows:

(trigger) (_on_) (ADDRESS) (__=__) (DELAY)

This enters the appropriate information into the state/software analyzer.

Press (execute) (RETURN)

The state/software analyzer will make the measurement and place a trace display on screen. The display will show the portion of trace memory that includes the trigger state.

STORING A SELECTED PORTION OF SOFTWARE ACTIVITY

You might want to examine just that portion of software that is involved with updating a display. If the display-update routine is located in the range of addresses from 3A3FH through 3B44H, and you grouped the probe lines that carry address information under the label ADDRESS, you can enter the following command line:

(store) (_on_) (ADDRESS) (__=__) (range) 313FH (thru) 3B44H

If you created a symbol map to support activity found on the address bus, and you included the symbol DISPLAY to identify the range of values from 3A3FH through 3B44H, then you could use that symbol in your command line, as follows:

```
store on ADDRESS = range DISPLAY
```

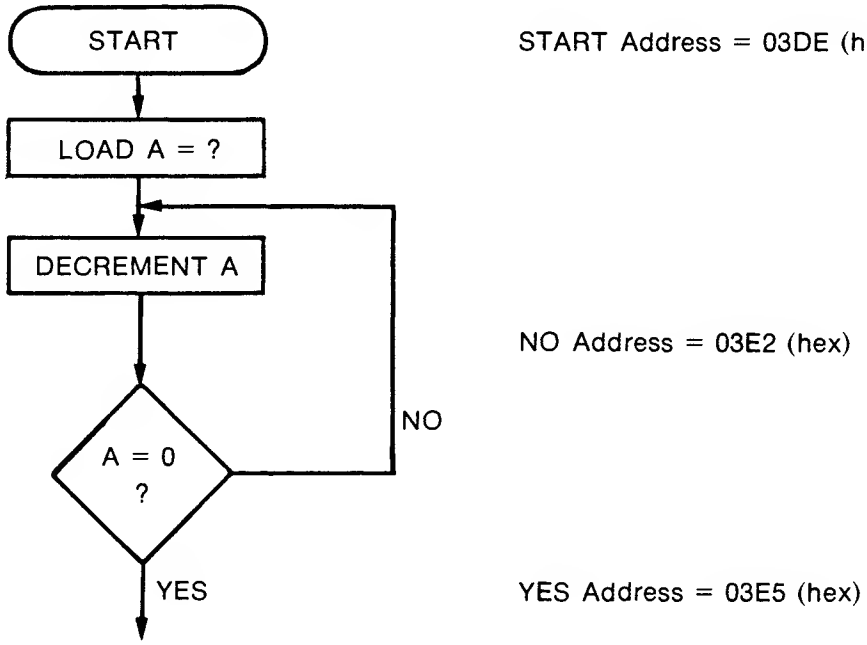
With either of the preceding command lines, the state/software analyzer will examine each state to see if it includes one of the specified values on the lines of the address bus. If it has an appropriate address value, the state will be stored in trace memory. If it does not have an appropriate address, the state will be ignored. At the end of the trace measurement, you can review the memory to see that it has stored only the activity involved with display updating.

NOTE

The trigger state will also be stored in the trace memory, even if it does not meet the store specification.

COUNTING USAGES OF A DELAY LOOP

You can use the probe lines connected to the address bus to count executions of a delay loop. Consider the example flow chart of a delay loop below:



To measure the executions within the above delay loop, set up the state/software analyzer as follows:

(trigger) (_on_) (ADDRESS) (__≡__) 03DEH

(store) (_on_) (ADDRESS) (__≡__) 03E5H

(count) (_on_) (ADDRESS) (__≡__) 03E2H

The measurement will begin when the analyzer recognizes START address 03DE (hex). It will count each 'NO' occurrence, address 03E2 (hex). It will store each 'YES' state, address 03E5 (hex), and along with that 'YES' state, it will store its present count of 'NO' states, and reset the counter to zero. At the end of the trace measurement, the memory will contain the trigger word and the results of 255 executions of the delay loop.

You can format a trace list to show information about the delay loop, as shown in the figure 8A-4.

Label:	ADDRESS	state count	state count
Base:	hex	rel	abs
Map:			
trigger	3DE	0	0
+001	3E5	99	99
+002	3E5	99	198
+003	3E5	99	297
+004	3E5	99	396
+005	3E5	99	495
+006	3E5	99	594
+007	3E5	99	693
+008	3E5	99	792
+009	3E5	99	891
+010	3E5	99	990
+011	3E5	99	1089
+012	3E5	99	1188

Figure 8A-4. Example Display of Delay Loop Activity

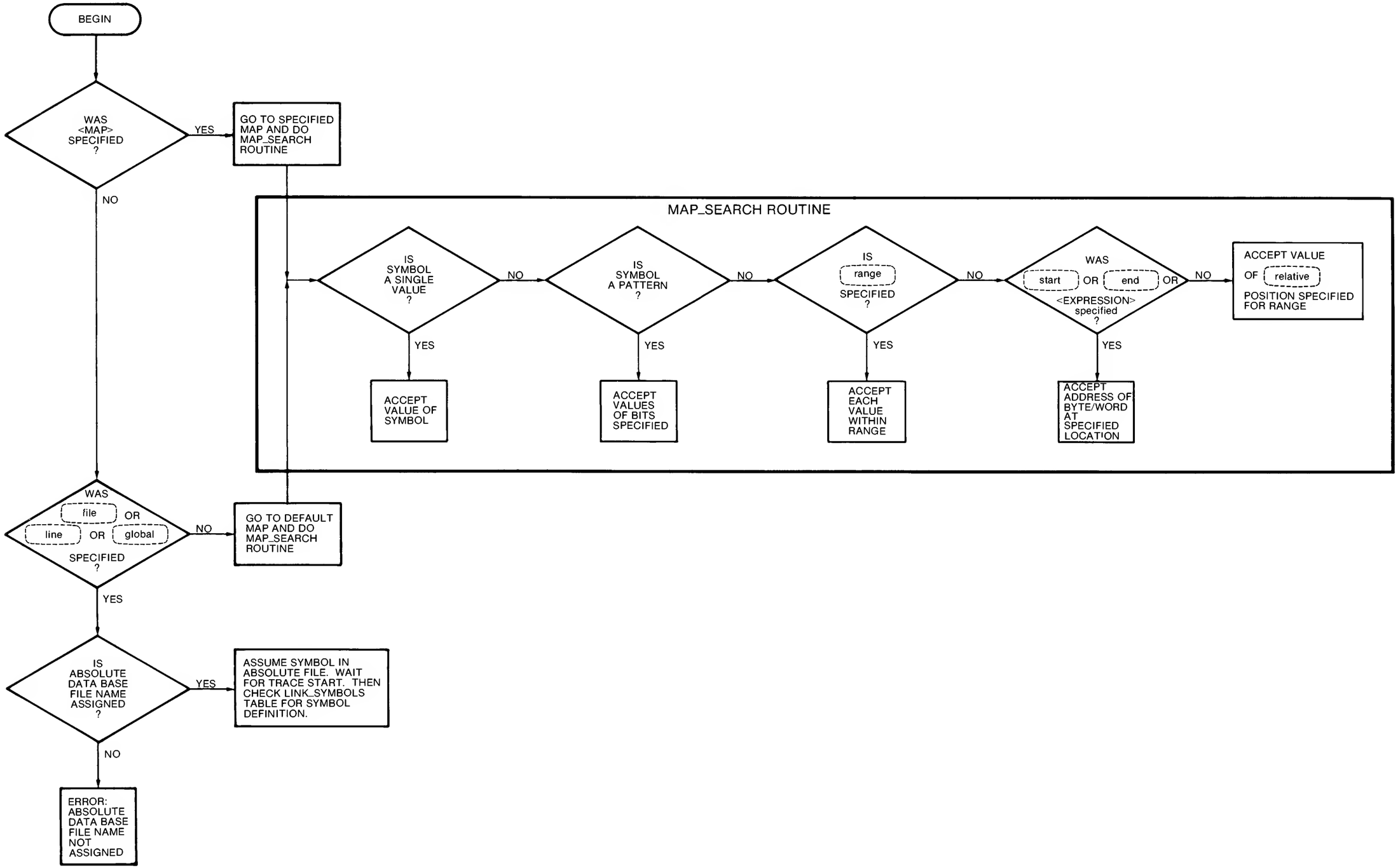


Figure 8A-5.
Symbol Search Before Trace Start

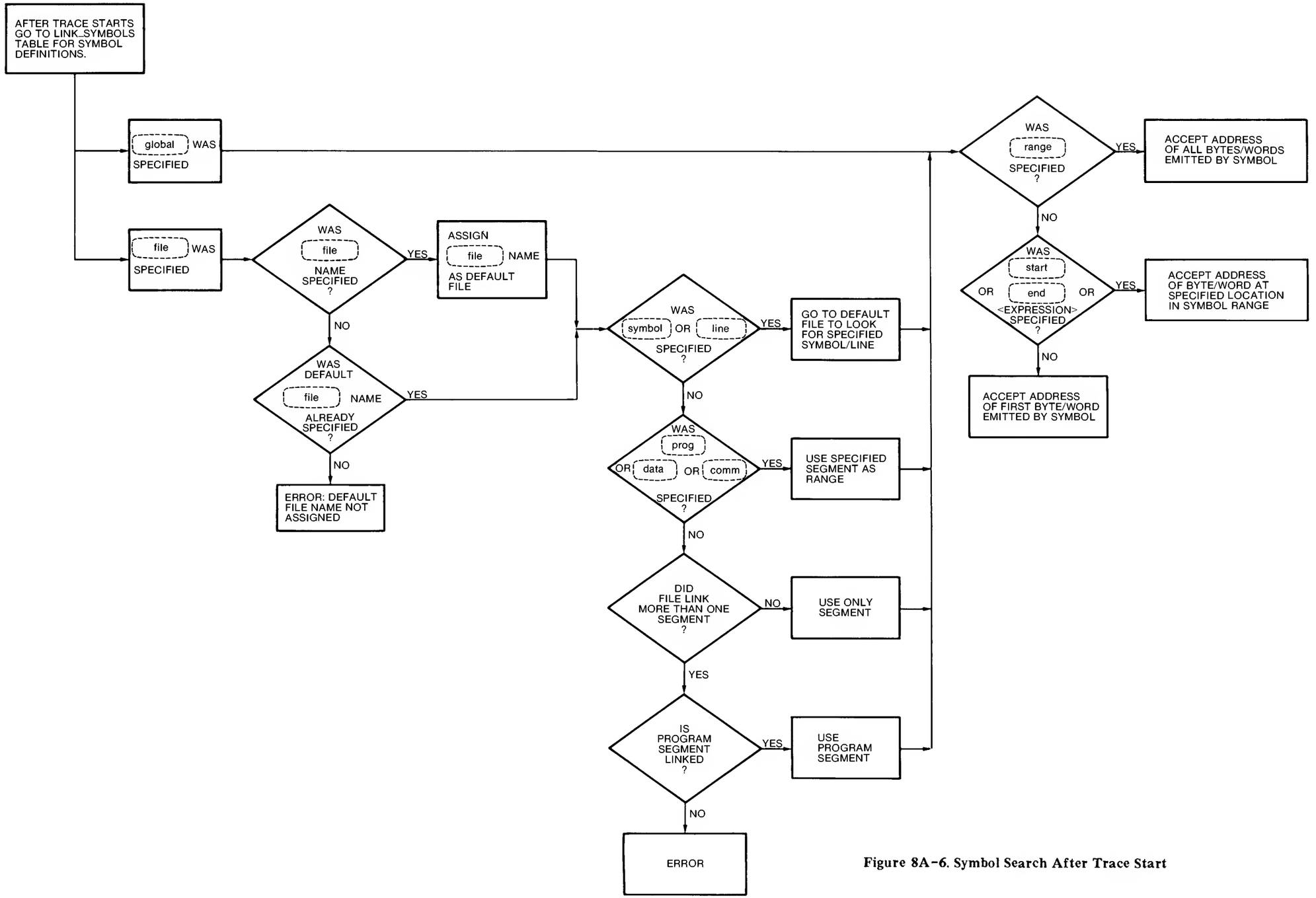


Figure 8A-6. Symbol Search After Trace Start

Chapter 8B

THE SEQUENCE AND WINDOW ELEMENTS

INTRODUCTION

The trigger, store, and count functions perform basic tracing. In the default condition of the trace specification, the trigger, store, and count specifications perform their functions without using the sequence and window elements. These elements can be used to obtain more complex specifications for the trigger, store, and count functions as well as control of the overview, assert, and master enable functions.

An example of the usefulness of sequencing is the ability to trigger the analyzer when a sequence of states is detected.

Example specification:

```
(sequence) (term_num) 1 (find) (ADDRESS) (==) 1000H
(sequence) (term_num) 2 (find) (ADDRESS) (==) 2000H
(sequence) (term_num) 3 (find) (ADDRESS) (==) 3000H (enable)
(trigger) (_on_) (sequence) (enable)
```

With the above specification, the trace memory will be filled with the states that follow the last sequence term. The trace point is more than a single pattern, as described in the basic tracing section. The trace point is an ordered flow of states (1000H, followed eventually by 2000H, followed eventually by 3000H).

This section describes how to use the sequence and window elements in trace specifications. There is one sequence element and two window elements. Each element functions independently from the other two. Each element provides an enable output which can be either true (enabled) or false (disabled). Each element can be set up to transition from enable to disable and transition from disable to enable at specific points in state or program flow. You can use the enable/disable level and the switching transitions from each element to control and qualify measurement parameters in the state/software analyzer.

The sequence and window elements always provide their assigned outputs whether the measurement parameters use them or not. The default state of these three elements is that they switch from disable to enable at the beginning of a trace and remain enabled throughout the trace.

This section will show you how to enter specifications for the sequence and window elements, and how to assign those elements to control the master, trigger, store, count, overview, and assert functions in the state/software analyzer. All of the symbol capabilities available for trigger, store, and count specifications (described in chapter 8A) are also available for use in the sequence and window specifications.

WHAT IS A WINDOW?

A window is a period in state flow where the state/software analyzer functions are enabled. The state/software analyzer functions are disabled during the portion of state flow that is outside the window. You can window your state flow measurements by slaving the analyzer measurement parameters to the sequence or window elements described in this section.

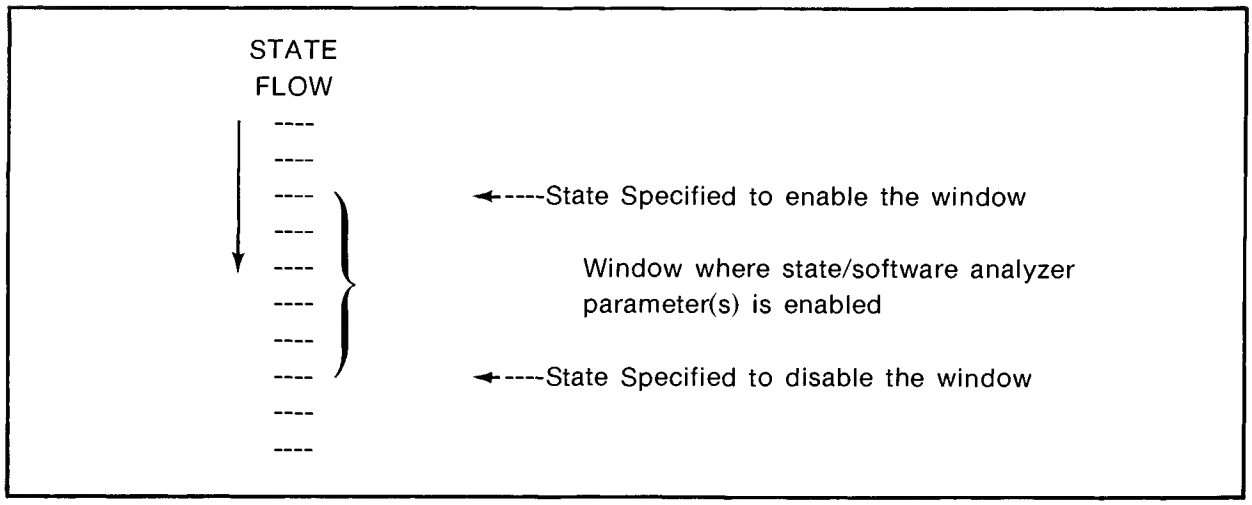


Figure 8B-1. Example Window

WHAT IS A WINDOW ELEMENT?

A window element is a component in the state/software analyzer that can produce three outputs: a transition when it switches from enable to disable, a transition when it switches from disable to enable, and an enable/disable window. This switching occurs at operator-selected points in state flow. See figure 8B-2.

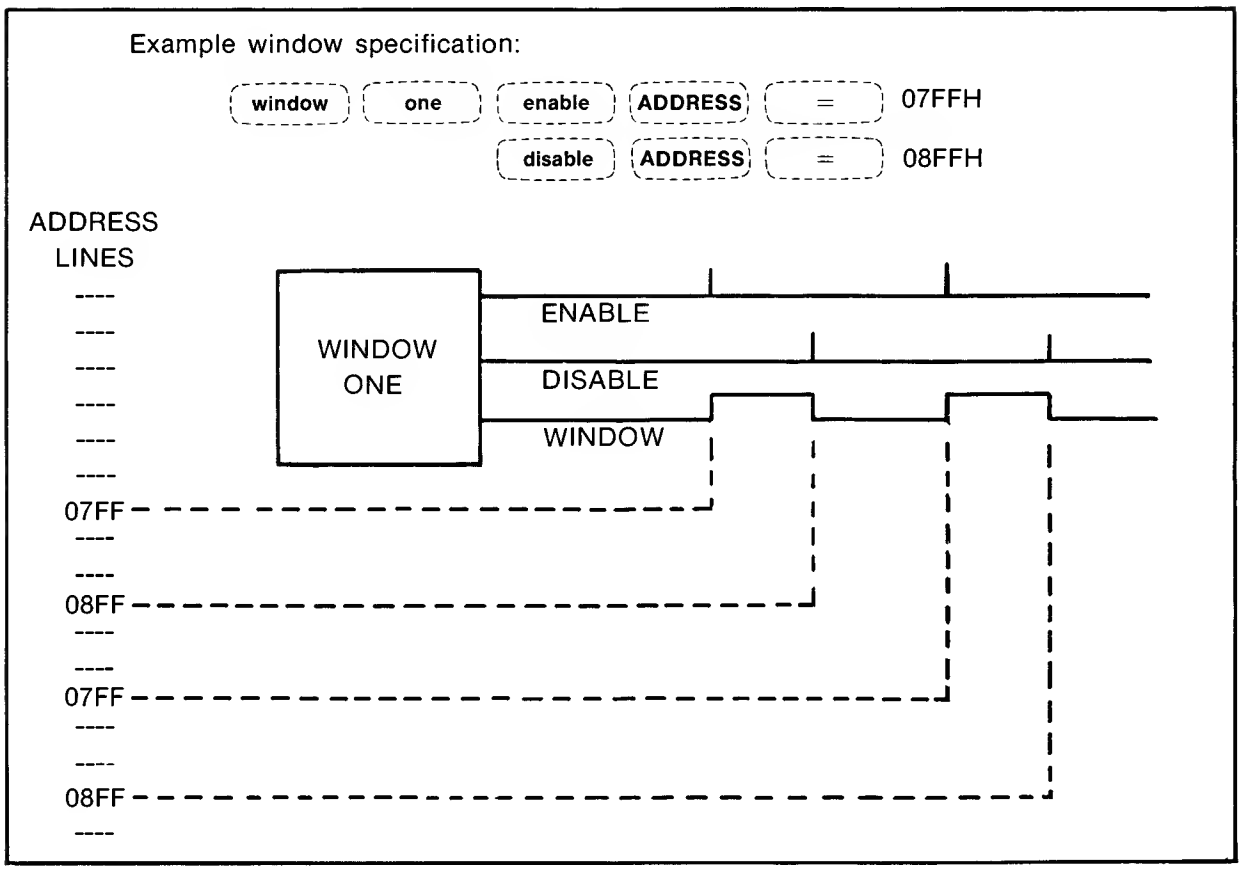


Figure 8B-2. Window Element

You can use the output from a window element to qualify or enable measurement parameters in the state/software analyzer. You can qualify trigger, store, and assert on the disable or enable transition. You can enable trigger, store, count, overview, overview count, and master enable during the enable period.

Window one is shown in the examples in this section. Window one and window two are identical in their operation.

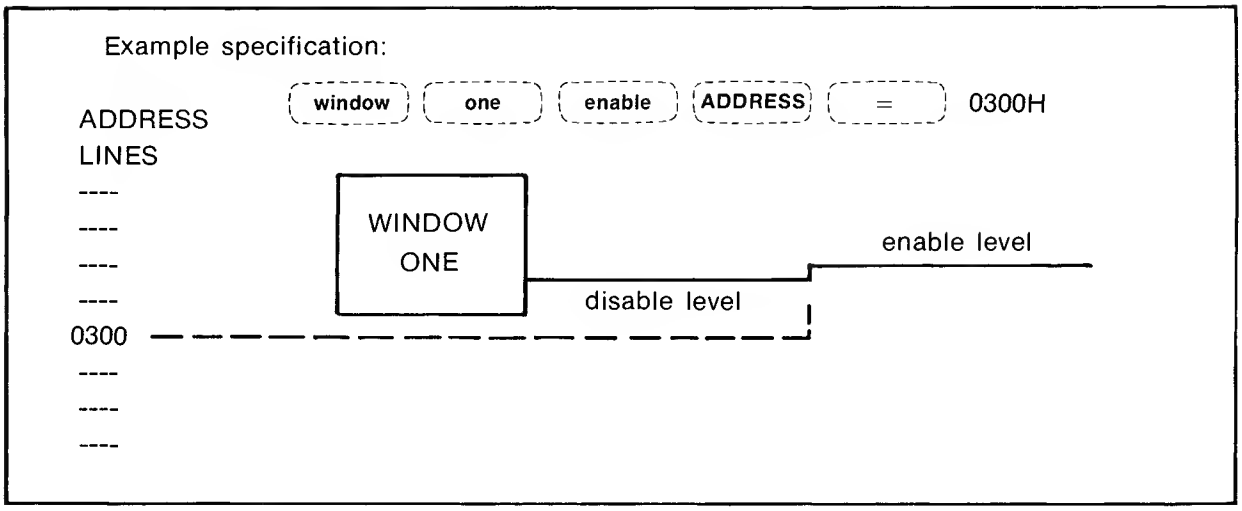


Figure 8B-3. Simple Enable Window

Figures 8B-3 and 8B-4 show simple enable and disable windows generated by window one. In figure 8B-3, window one begins in the disable condition because it is specified to enable when it finds 0300H on the address lines. When it finds the specified enable address, it switches to the enable condition and stays there for the remainder of the measurement.

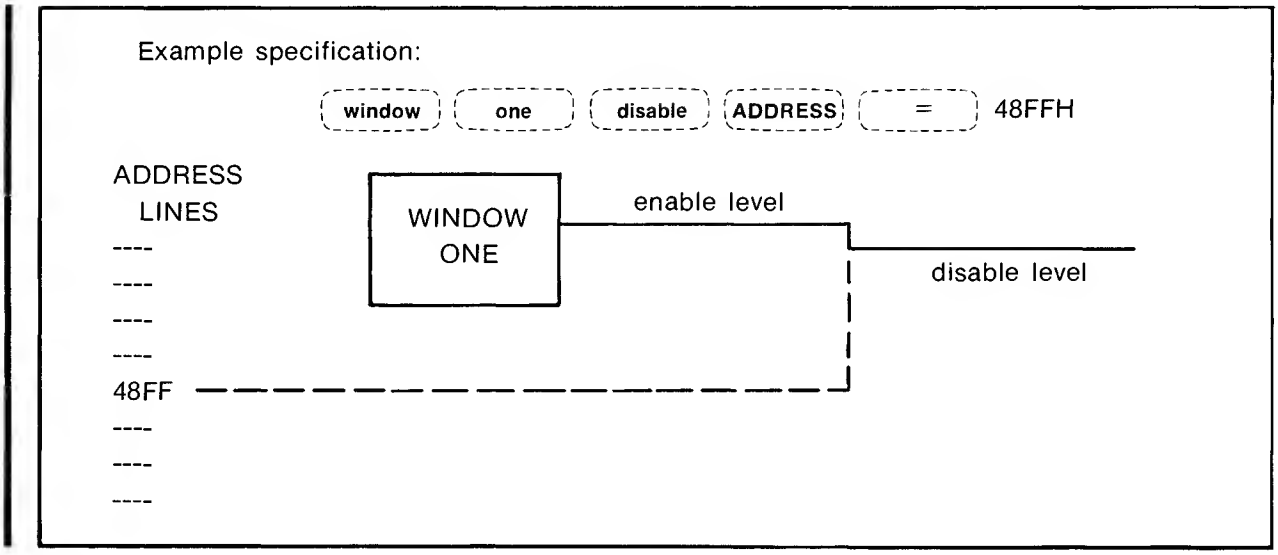


Figure 8B-4. Simple Disable Window

In figure 8B-4, the window element begins in the enable level because it is specified to disable when it finds 48FFH on the address lines. When it finds the specified address, it switches to the disable condition and stays there for the remainder of the measurement (similar to simple enable window).

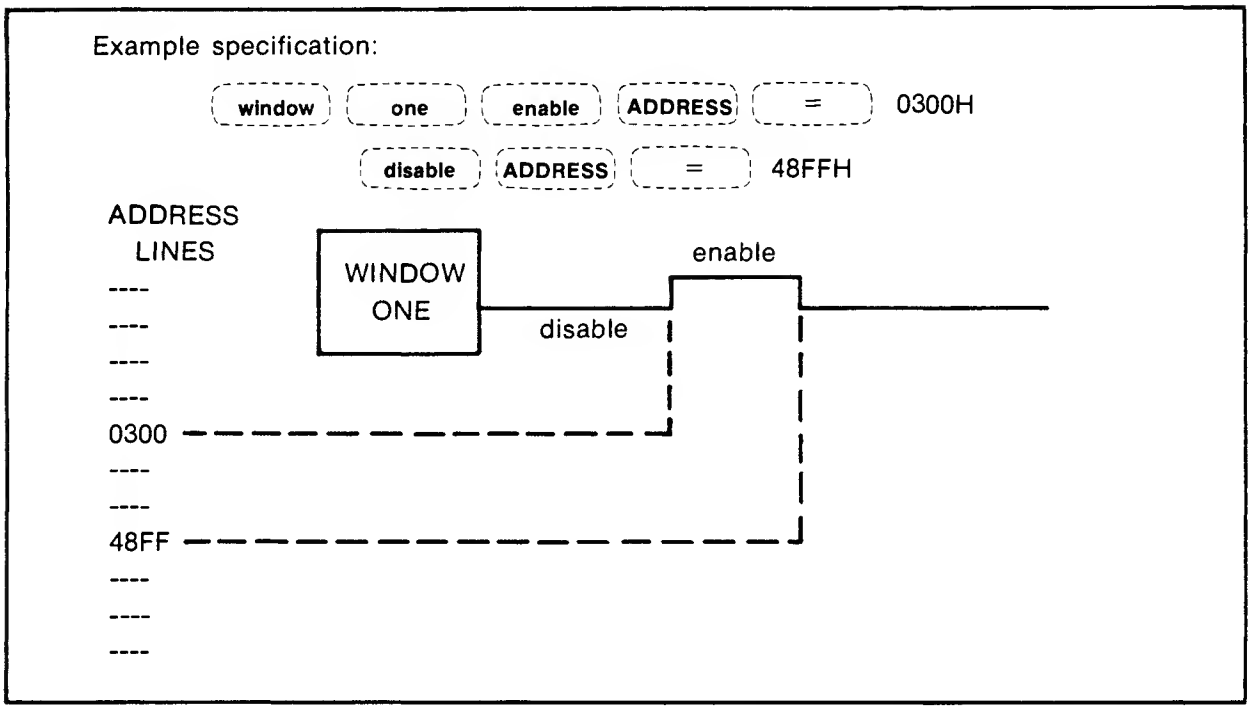


Figure 8B-5. Enabling Window

Figure 8B-5 shows an enabling window. When the measurement begins, the window element is in the disable condition because its first instruction is a switch-to-enable state. When the enable address is found on the address lines, the window element will switch to the enable condition. When the disable address is found on the address lines, the window element will switch to the disable condition. The window element will continually search for the state that will cause it to switch. This switching will continue as long as the measurement is in progress.

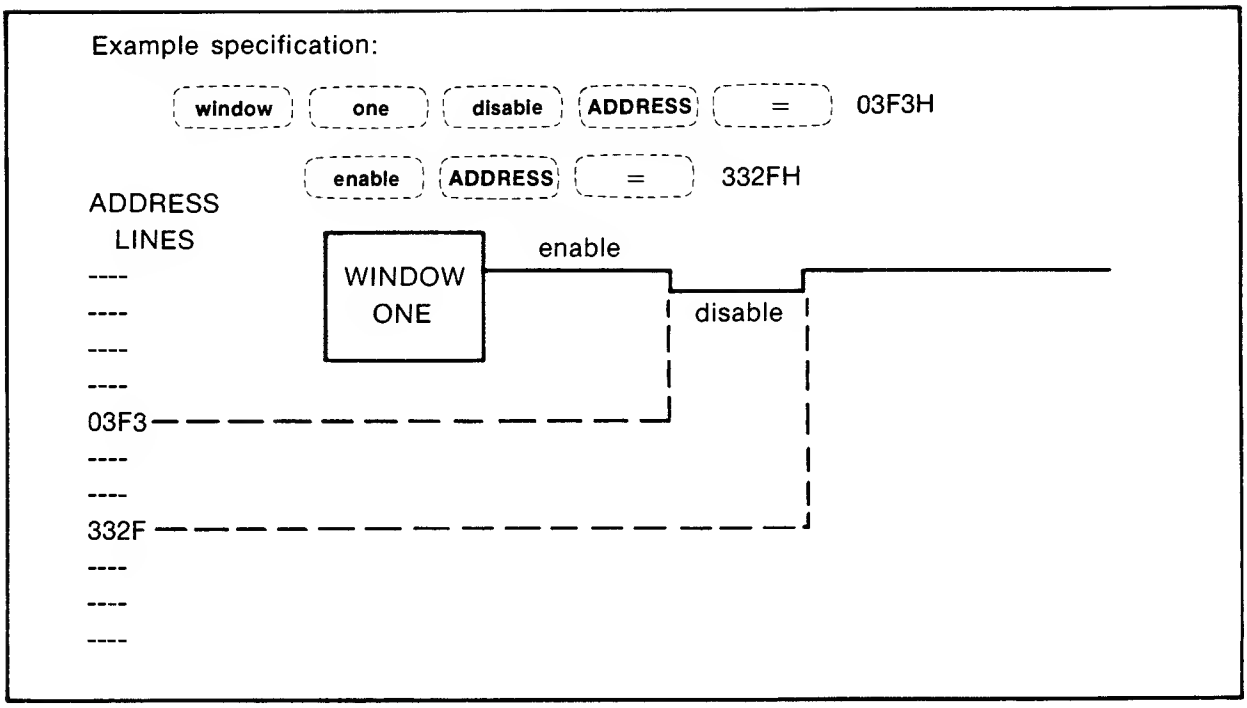


Figure 3B-6. Disabling Window

Figure 8B-6 shows a disabling window. When the run begins, the window element is in the enable condition because its first instruction is a switch-to-disable state. The output of the window element switches between disable and enable as the appropriate states are found (similar to enabling window).

DIFFERENCES BETWEEN SEQUENCE AND WINDOW ELEMENTS

The sequence element provides the same kind of output as a window element. It is either "enable" or "disable". But the sequence element can accept specifications that are more extensive than the window elements.

A window element can only accept a point for switching from disable to enable, and a point for switching from enable to disable.

The sequence element can accept a series of points to be found in a specified order before it switches its output. See figure 8B-7.

Example Sequence Specification:

```
sequence  term_num  1  find  ADDRESS  =  1000H
sequence  term_num  2  find  ADDRESS  =  2000H
sequence  term_num  3  find  ADDRESS  =  3000H  enable
sequence  term_num  4  find  ADDRESS  =  4000H
sequence  term_num  5  find  ADDRESS  =  5000H  disable
```

The sequence element will produce the following output:

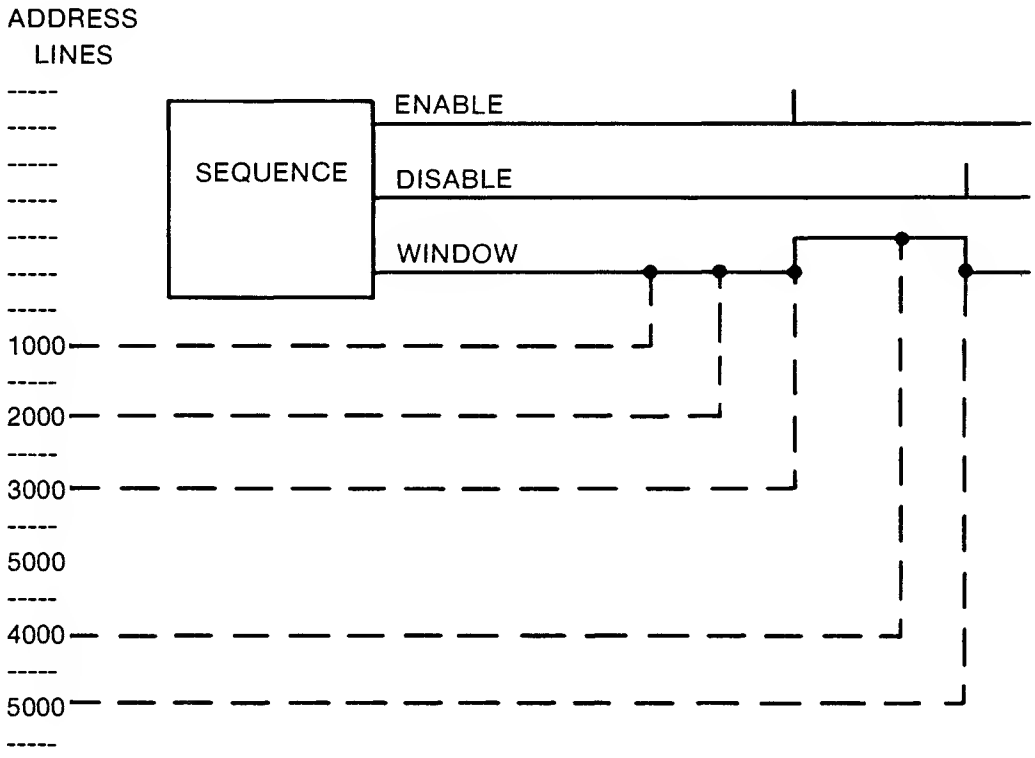


Figure 8B-7. Sequence Element

In figure 8B-7, the sequence element must find the three terms in the order specified before it will change its output from disable to enable. Because of the way the specification was stated, these three terms can be consecutive, or separated by many other states in the program flow, and the specification will still be satisfied, as long as the terms are found in order.

Terms 4 and 5 must be found in order before the sequence element can switch its output from enable to disable. Address 5000 did not cause the sequence element to change its output the first time it occurred because address 4000 had not been found before it. After address 4000 was found, then address 5000 did cause the sequence element to change its output.

SPECIFYING AN OCCURRENCE COUNT

You can specify that states must be found more than one time before the sequence element can look for the next term, as follows:

```
[sequence] [term_num] 1 [find] [ADDRESS] [  =  ] 1000H
        [occurring] { [immediate] } 10 [times]
                    { [eventual ] }
```

If you select [immediate], 1000H must be found on the ADDRESS lines ten consecutive times. If found less than 10 times, the count of 1000H resets to 0 and starts over.

If you select [eventual], 1000H need only be found ten times. These do not need to be consecutive occurrences.

You can also specify that two or more different states must be found in order to satisfy a sequence term, as follows:

(sequence) (term_num) 1 (find) (ADDRESS) (==) 1000H

(followed) { (immediate) / (eventual) } (ADDRESS) (==) 2000H

If you select (immediate), 2000H must be the next state to appear after 1000H. If not, the sequence element begins searching for the next occurrence of 1000H.
If you select (eventual), 2000H need only appear as one of the states following 1000H.

You can use a combination of (occurring) and (followed) to specify a sequence of states which must occur a number of times to satisfy the term, such as:

(sequence) (term_num) 1 (find) (ADDRESS) (==) 1000H

(followed) (immediate) (ADDRESS) (==) 2000H

(followed) (immediate) (ADDRESS) (==) 3000H

(occurring) (eventual) 5 (times) (enable) (RETURN)

You can also include a "restart" term in your enable sequence and/or your disable sequence. Figure 8B-8 shows a restart term in the sequence specification. When a restart term is found in program flow, the state/software analyzer will return to the beginning of the sequence that it was trying to satisfy and start looking again for the first term in that sequence.

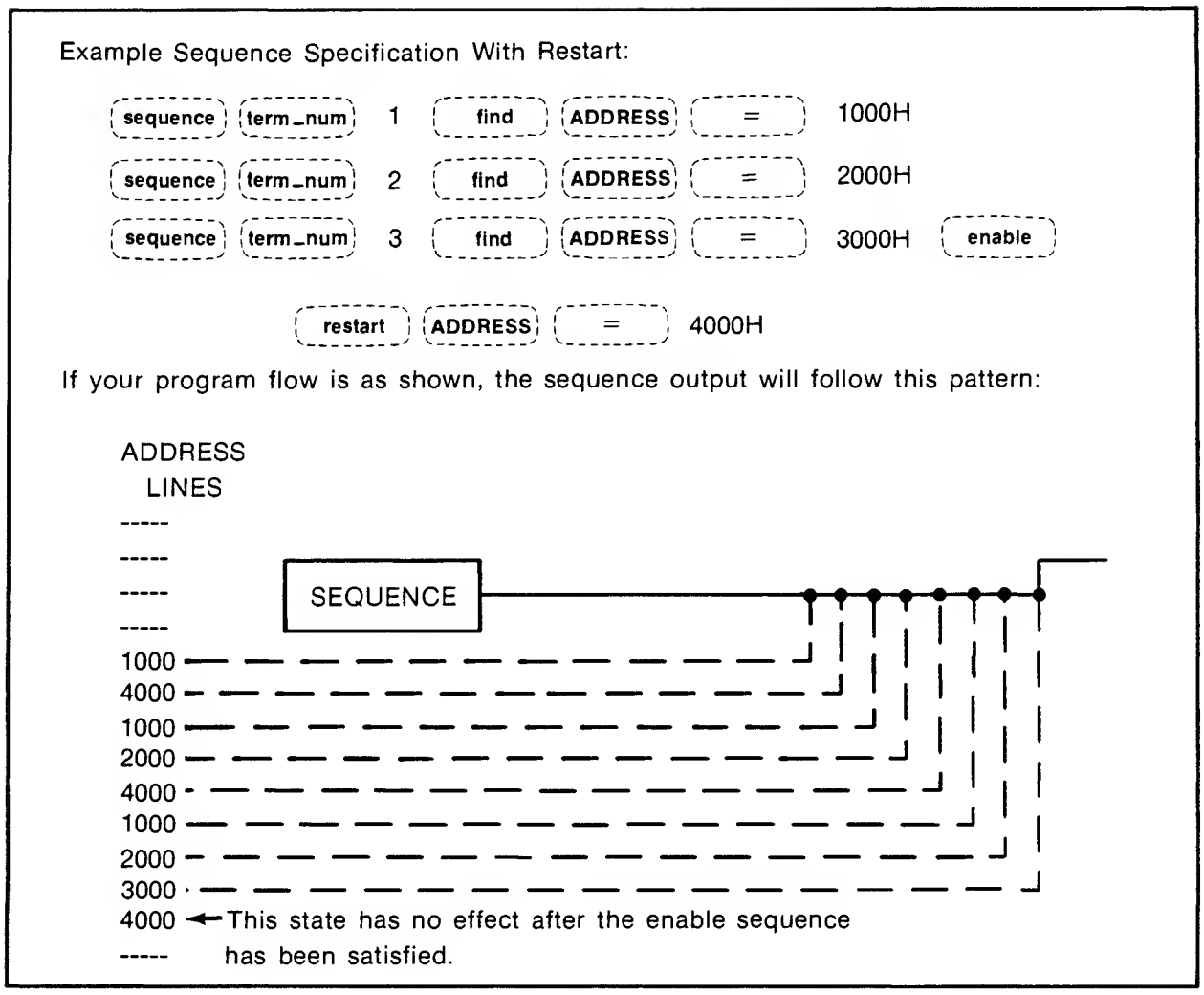


Figure 8B-8. Using A Restart In Your Sequence

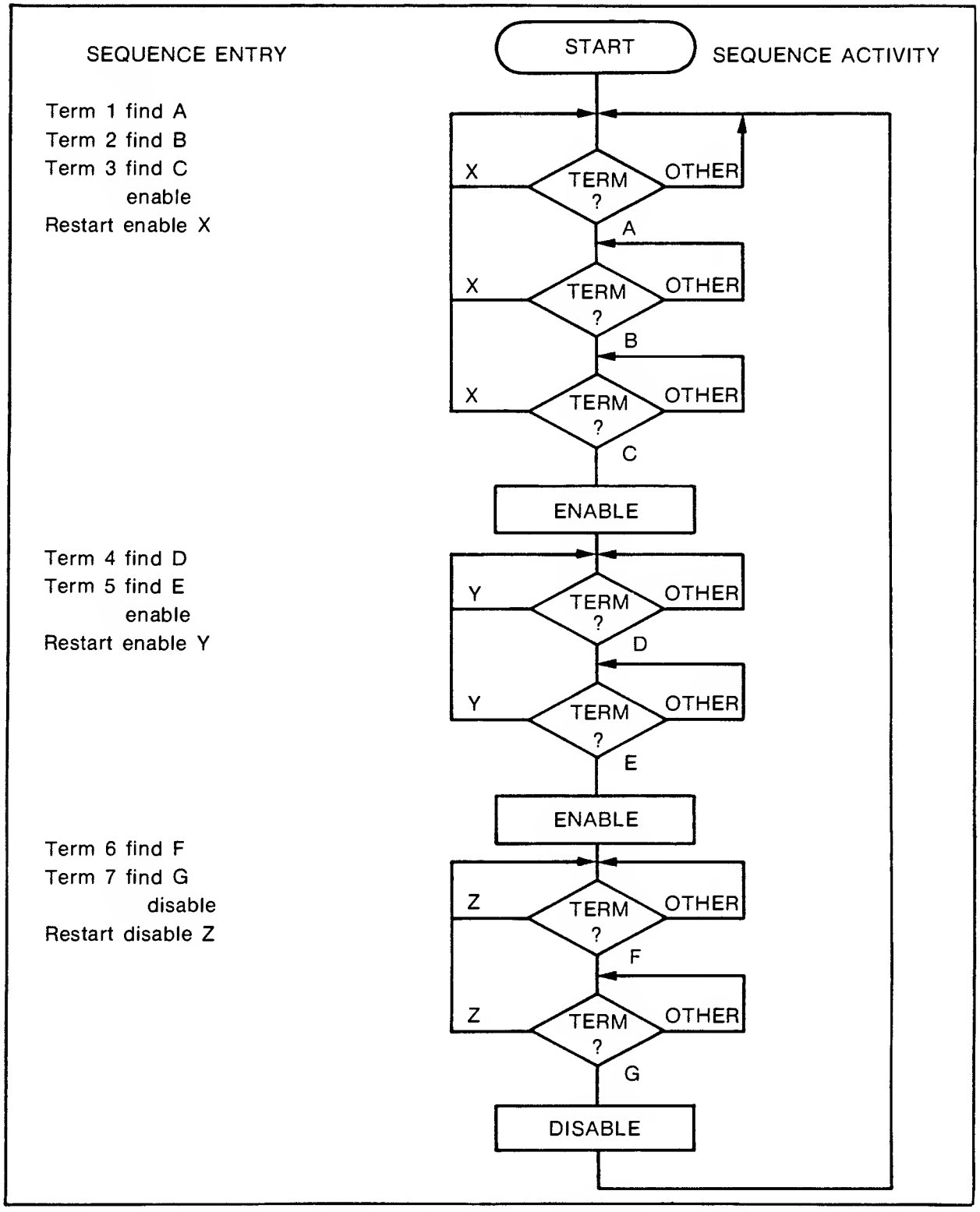


Figure 8B-9. Sequence Flow Chart

The flow chart in figure 8B-9 shows the special capabilities of the sequence element. The following paragraphs describe features in the flow chart that you need to understand before you can use the sequence element to its full capability.

a. Once a trace begins, the analyzer examines each new state in its search for sequence terms. This search continues throughout the entire trace. If a state meets the requirements of the advance term, the sequence element begins looking for the next advance term. If the state meets the requirements of the restart term, the sequence element returns to the last enable or disable point and begins a new search for the first sequence term. If the state does not meet the requirements of either the advance term or the restart term, no change occurs in the sequence and the analyzer waits to examine the next state.

b. When a "restart" term is found, the search for sequence terms returns to the last "enable" or "disable" term, not to START, except in the first enable/disable sequence.

c. You can specify multiple "enable" sequences before a "disable" sequence. You can also specify multiple "disable" sequences before an "enable" sequence. Multiple "enable" sequences are useful if you want the analyzer to generate a series of pulses on BNC-port-1 and/or the stimulus line (refer to Chapter 9). You can also use multiple enables to measure relative time between a series of events in program flow. The store command can specify multiple enables as follows: (store) (on) (sequence) (enable).

NOTE

Multiple enable and disable occurrences are not available in the specifications for window elements.

d. In the example of figure 8B-9, the sequence flow wraps around because the sequence ends in the same condition that it began. If you enter a sequence specification that ends with the sequence element in the opposite condition from where it began, your sequence element will not wrap around and repeat.

NOTE

The analyzer displays a bright bar in the trace specification to indicate sequencer status. The bright bar is shown beside the sequence term that the sequence element is presently searching for. This is useful when a trace that includes a sequence specification fails to perform as expected. The bright bar shows which sequence term has not been found.

SEQUENCE AND WINDOWS IN COMBINATION

Once you make specifications for the sequence, window one, and/or window two, their outputs become resources. As resources, they can be used to window portions of state flow, enable and disable measurements, and/or qualify data acquisition and trace parameters. See figure 8B-10. You can slave the analyzer trace parameters to the window periods and/or to the enable and/or disable transitions. Because you can assign different specifications to each of the three elements, you can have up to three different windows and six or more enable/disable transitions for controlling measurement parameters.

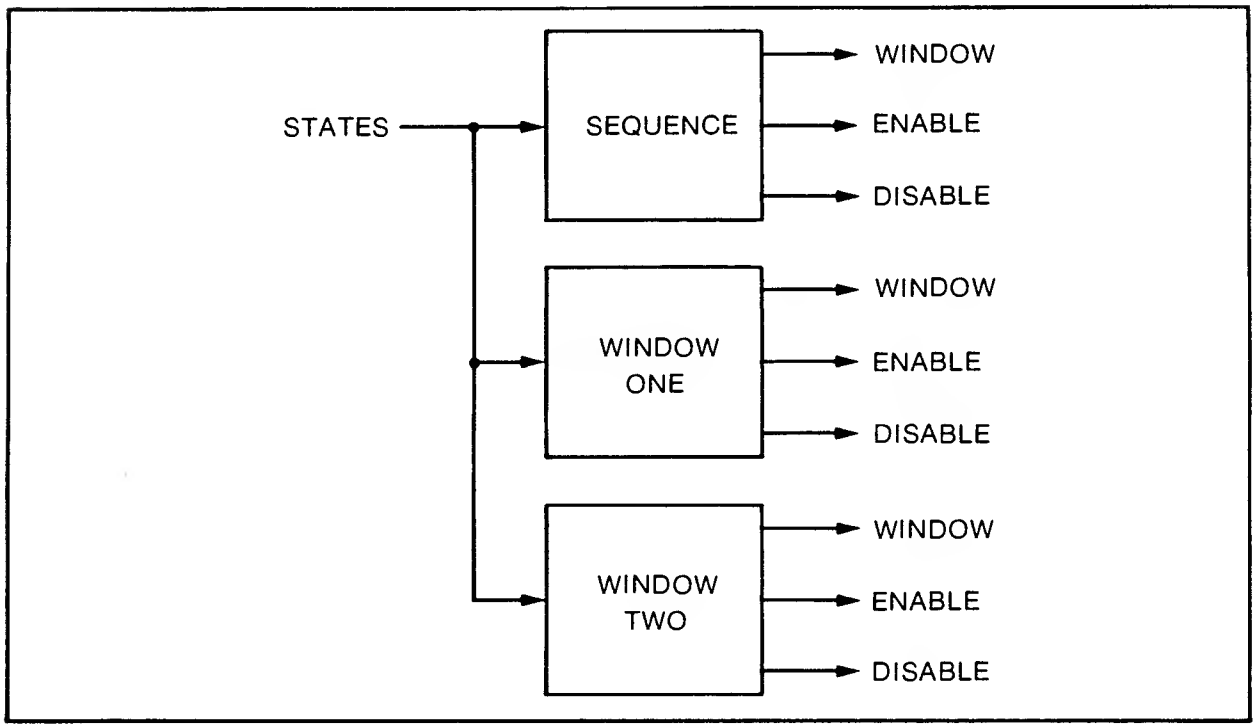


Figure 8B-10. Sequence And Window Resources

ALLOCATING SEQUENCE AND WINDOW OUTPUTS

The sequence, window one, and window two elements each have three outputs: window, enable transition, and disable transition. With these nine outputs, you can enable and qualify your measurement parameters. Figure 8B-11 shows the way that the sequence and window outputs are made available to the seven parameters.

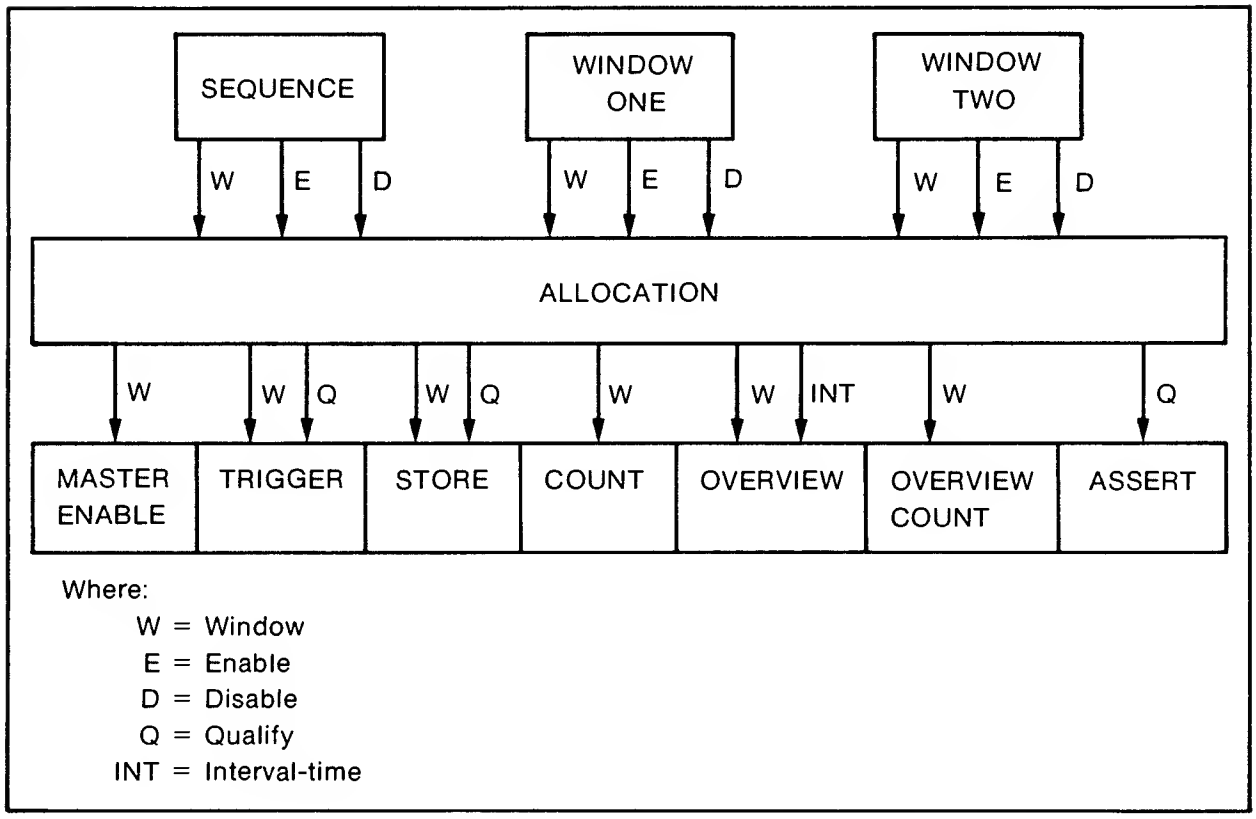


Figure 8B-11. Allocating Sequence And Window Outputs

Figure 8B-12 shows a very simple allocation of two of the sequence outputs. The window output is used to enable storage of states (during the enable period). The enable transition is used to qualify triggering at the start of the states associated with the sequence window.

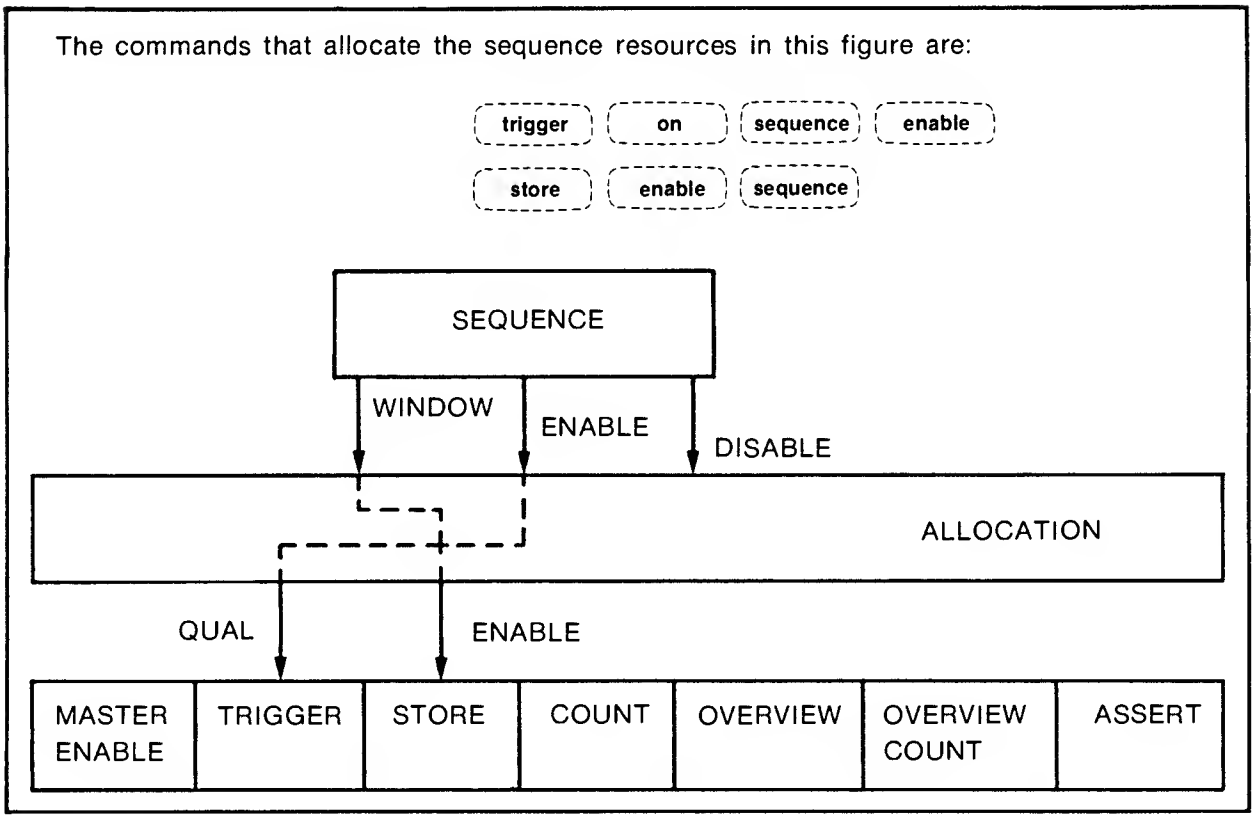


Figure 8B-12. Sequence Outputs Allocated To Trigger And Store

Figure 8B-13 shows the ways that the seven measurement parameters of the state/software analyzer can use the sequence and window outputs. The output from a single sequence or window element can be used by any number of these measurement parameters as shown in figure 8B-15.

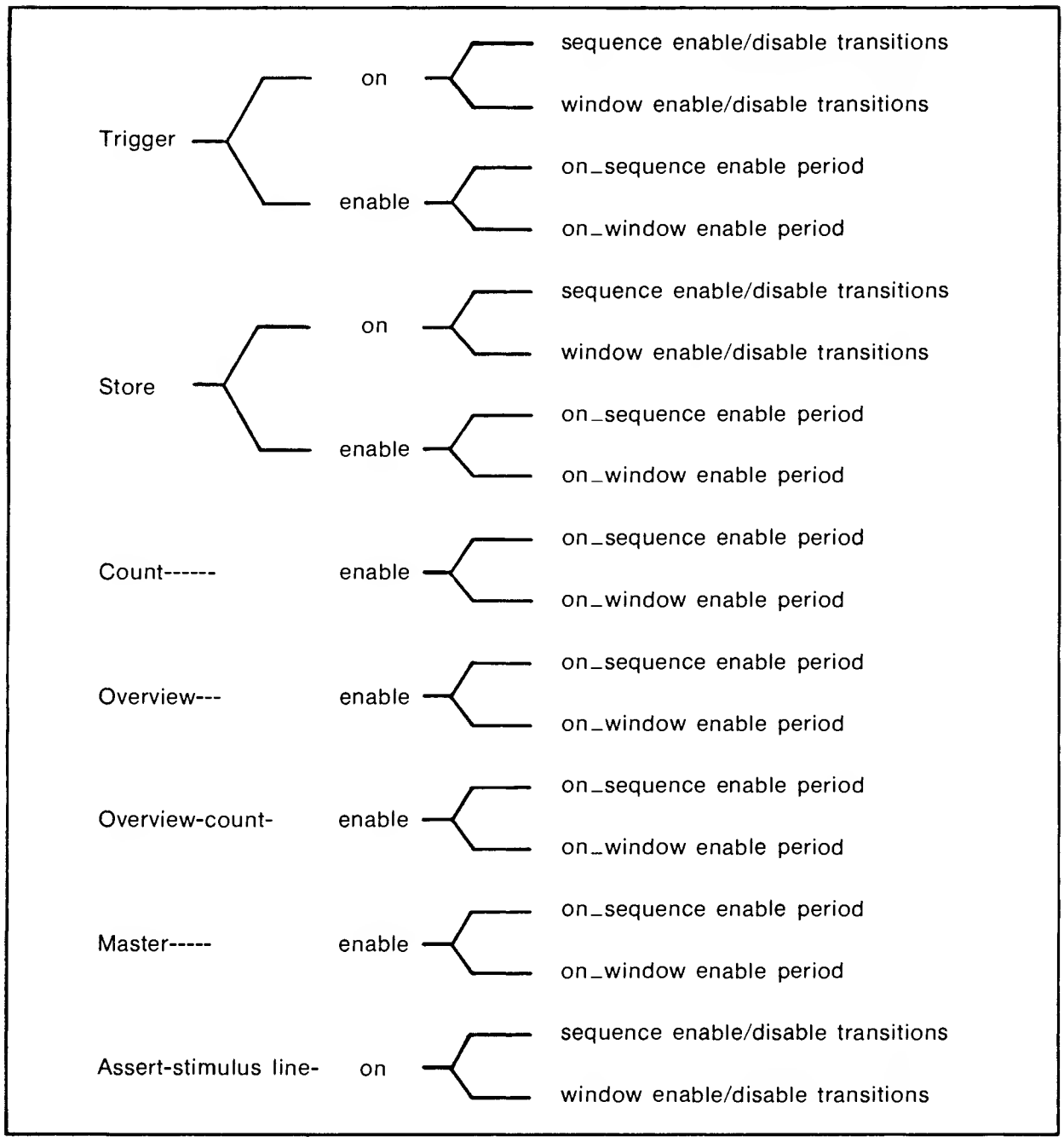


Figure 8B-13. Allocation Of Sequence And Window Outputs

When the trace functions (trigger, store, and count) are enabled by a sequence or window element, they become enabled following the state that caused the enable transition.

Example: (trigger) (enable) (sequence)

The state which causes the sequence to switch to enable is not included in the window of states which can be identified as the trigger.

When performing overview on a ranging label, and a sequence or window element is used to enable the overview functions, the state that causes the enable transition in the window or sequence is included in the measurement.

Example: (overview) (enable) (sequence)

The state which causes the sequence to switch to enable will be captured in the overview event memory.

When performing interval overview measurements (overview on time count, or overview on state count), the overview enable defines the start/stop interval. A detailed description of interval overview measurements is provided in the overview measurements chapter.

Functions which are directly qualified by sequence or window transitions or terms (such as trigger on sequence enable or assert stimulus on window one disable) are active on the state which causes the transitions, or satisfies the terms.

Example: (store) (_on_) (sequence) (terms_dis)

The states which satisfy terms leading up to an enable transition in the sequence will be stored.

SHARING SEQUENCE/WINDOW TERMS

You can create up to three different enable/disable windowing activities for measurement control, using the sequence and window elements. These allow you to create complex hierarchy for the capture of selected software activity. There are 15 terms available to be shared among the sequence, window one, and window two elements. These terms can be shared in one of the three following ways:

1. All 15 terms can be allocated to the sequence element. They can be used to create a long series of terms for enable and disable qualification. Window one and window two are not used in this case.

EXAMPLE:

SEQUENCE	WINDOW ONE	WINDOW TWO
1 find ...	off	off
2 find ...		
3 find ...		
4 find ...		
5 find ...		
enable		
6 find ...		
7 find ...		
8 find ...		
9 find ...		
enable		
10 find ...		
11 find ...		
12 find ...		
13 find ...		
14 find ...		
15 find ...		
disable		

2. Seven of the terms can be allocated to the sequence element to create a complex qualification set, and one of the windows can also be activated with terms of its own. The other window is not used.

EXAMPLE:

SEQUENCE	WINDOW ONE	WINDOW TWO
1 find ...	find ...	off
2 find ...	enable	
3 find ...	find...	
4 find ...	disable	
enable		
5 find ...		
6 find ...		
7 find ...		
disable		

3. Three terms are allocated to the sequence element to create a simple qualification set. Both of the window elements are also activated with terms on which to generate their window, enable, and disable outputs.

EXAMPLE:

SEQUENCE	WINDOW ONE	WINDOW TWO
1 find ...	find ...	find ...
2 find ...	enable	enable
enable	find ...	find ...
3 find ...	disable	disable
disable		

NOTE

In this case, the sequence specification cannot include a "restart" term.

EXAMPLES

USING SEQUENCE TO ENABLE TRIGGER

You might want the state/software analyzer to trigger a trace after the first occurrence of 04FF (hexadecimal) on the address bus. You could simply enter the following trigger specification:

(trigger) (_on_) (ADDRESS) (___=___) 04FFH

You might want this trigger to occur only when 04FF follows 33A7. The basic trigger specification will not accept this sort of entry, but you can use the sequence element to accomplish this trigger, as follows:

```
(sequence) (term_num) 1 (find) (ADDRESS) (__=__) 33A7H (enable)

(trigger) (enable) (sequence)

(trigger) (_on_) (ADDRESS) (__=__) 04FFH
```

The above specification slaves trigger enable to the sequence element. The sequence starts off in the disable condition. Each state is examined for an address of 33A7 (hex). When that address is found, the sequence will switch to the "enable" condition. This switching transition will be recognized by the trigger, enabling it to begin looking for its trigger event (04FFH).

USING A WINDOW ELEMENT TO CONTROL STORE

```
(window) (two) (enable) (ADDRESS) (__=__) 0400H (disable)

    (ADDRESS) (__=__) 0410H

(store) (enable) (_on_) (window) (two)

(store) (_on_) (any_state)
```

The above command lines slave the store specification to window two. When the trace begins, window two is in the disable condition. It examines each state, but no state is stored in memory. When window two finds 0400H on the address lines, it switches to the enable condition. The enable condition is recognized by the store function. Now the store function begins storing states that match its store qualification (store any_state, in this example).

Window two now examines each state until it finds 0410H on the address lines. At this point, it switches back to the disable condition. The disable condition is recognized by the store function, and it stores no more states in memory.

This activity continues for the remainder of the trace. Each time 0400H appears on the address bus, window two enables and the store function starts saving states in memory. Each time 0410H appears on the address bus, window two disables and storage stops. By using the window element, you can obtain information that will be missed by entering a single range of addresses into the store specification. If you are examining a routine that reads parameters from stack space, the read activity will be captured in memory when you use the window because store is enabled until 0410H is found. It would be inconvenient to use the store specification by itself to capture this information. You would have to include in your store specification every range of addresses that might ever be accessed by this routine. For example, the following command:

(store) (_on_) (ADDRESS) (__=__) (range) (STACK)

This would cause the analyzer to store activity with addresses in the range of the symbol STACK. Any activity executed during stack operations with addresses outside the range of the symbol STACK would not be stored in memory.

USING A WINDOW ELEMENT TO CONTROL TIME MEASUREMENTS

You might like to make time measurements between specific points in a program or state flow. The following command line would give you time measurements between every state stored in memory:

(count) (_on_) (time)

What if you wanted only to measure the time between 0403H and 0409H. The above entry will make a measurement of time between every state. After the measurement, you will have to search the entire memory to find the address lines 0403H through 0409H. You will have to display the absolute time count and subtract one of the time counts from the other to find the time between them. By store qualifying only the two events of interest, this work will be eliminated.

(store) (_on_) (ADDRESS) (__=__) 0403H

(_or_) (ADDRESS) (__=__) 0409H

To further condense the information stored in the trace memory, a window element can be used to enable the time count and store only at the end of each interval:

```
(window) (_one_) (enable) (ADDRESS) (__≡__) 0403H (disable)

    (ADDRESS) (__≡__) 0409H

(count) (enable) (window) (_one_)

(store) (_on_) (window) (_one_) (disable)
```

With the above entries, the count function is controlled by window one. When the measurement starts, window one is in the disable condition. When 0403H is found on the ADDRESS lines, window one switches to enable. This starts the counter. The counter measures time for the period of the enable condition. When window one finds state 0409H, it switches to disable. This stops the counter. Storage is specified to occur on window one disable so this state, along with the total time count, will be stored in trace memory. Then the counter will be reset. This process will be repeated until the trace memory is full.

USING A WINDOW ELEMENT TO CONTROL OVERVIEW

You can make overview measurements on a labeled set of probes without using a window element. You might set up the state analyzer with the following example commands:

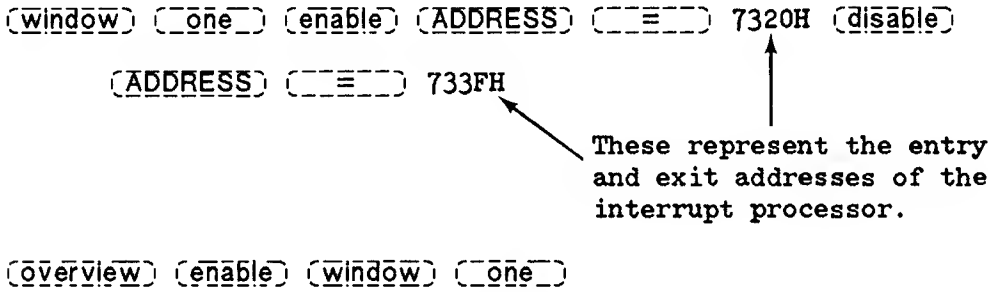
```
(overview) (_on_) (ADDRESS)

(overview) (event) 1 (is_range) 0 (thru) 100H

(overview) (event) 2 (is_range) 101H (thru) 200H

(overview) (event) 3 (is_range) (else)
```

These commands will cause the state/software analyzer to store an event for each state in the program flow. What if you want to overview the activity during the interrupt processing routine alone. Use a window element, as follows:



(overview) (enable) (window) (_one_)

With the above entry, the overview measurements will be made only during the enable period from window one (the interrupt processor routine). Each time an interrupt process is started, event capture will begin. When the interrupt process is completed, event capture will be halted, and it will remain halted until the interrupt occurs again. By using the window element, all of the overview events will represent states found between entry to and exit from the interrupt processor routine.

Chapter 8C
OVERVIEW MEASUREMENTS

INTRODUCTION

Overview measurements provide a higher level for viewing software operation than is available by capturing the details of state transactions. You can use overview measurements to compare relative performance of software routines. You can use these measurements to characterize process flows and execution times. You can also measure the usage of resources in a system under test.

As an example of the usefulness of overview measurements, see how the relative usage of the RAM and ROM service routines can be compared.

Example Specification:

```
(overview) (_on_) (ADDRESS)

(overview) (event) 1 (named) ROM (is_range) 4000H (thru) 5600H

(overview) (event) 2 (named) RAM (is_range) 6000H (thru) 64FFH
```

In the example, the overview memory will be filled with events captured from activity on the ADDRESS bus. Each time an address between 4000H and 5600H is found, the overview measurement will store the code for event 1. Each time an address is found that is between 6000H and 64FFH, the overview measurement will store the code for event 2. All other addresses will be disregarded. At the end of this measurement, the analyzer can display a histogram that shows the actual number of event 1 and event 2 transactions that were measured, along with bright bars whose lengths are proportional to the amount of memory occupied by each event (ROM and RAM).

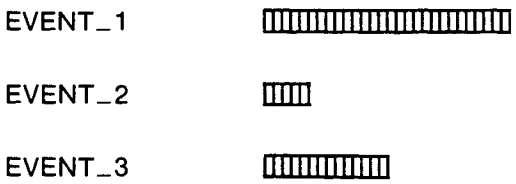
This chapter describes the three modes of making overview measurements: overview on a ranging label, overview on state count, and overview on time count. Then this chapter gives detailed information about how to specify overview events, and how the event detector works. At the end of the chapter are several examples showing usage of the overview measurement capabilities.

In an overview measurement, the state/software analyzer can detect and store over four-thousand separate overview events. The overview memory stores a representation of each overview event as it is detected. An overview display of program activity shows information about these collected events. The information can be presented in the form of histograms, graphs, or lists.

The state/software analyzer can search for up to 15 different overview events at the same time. Each event is assigned a name. The name will be either a name you entered using the named softkey, or it will be a restatement of the specification assigned to the event. The names are shown on the display. Beside each event name are numbers that show how many of the locations in the overview memory stored codes representing each event. You can also have the state/software analyzer store the "everything else" event. If you specify its storage, this event will be stored each time the measured data does not meet the specifications of any of your named events.

There are three different overview measurement modes. They all acquire events into memory in about the same manner. A 20-bit code is sent to the event detector. The event detector compares the 20-bit code with the event specifications that were set up before the start of the measurement. Each event has a unique 4-bit code assigned to it. When the event detector finds the specifications met by the 20-bit input code, it outputs the corresponding 4-bit event code. The 4-bit event code is stored in the overview memory.

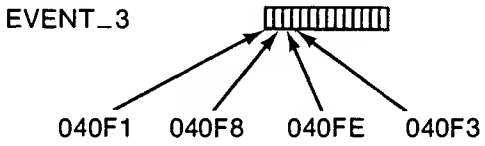
You must understand the difference between state measurements and overview measurements before you can use the overview measurements to advantage. Figure 8C-1 illustrates this difference.



The above is a typical histogram display of three events on the ADDRESS label that were monitored during a measurement. EVENT_3 will be examined.

If the specification for EVENT_3 is the range of states 040F0H through 040FFH, then any value that falls within that range will result in the storage of the code for EVENT_3 in overview memory.

The actual values that were detected during the storage of EVENT_3 might have been as shown below:



There is no way to determine which specific values caused the storage of EVENT_3 codes; only that they did meet the specification for EVENT_3 (some value from 040F0H to 040FFH).

Figure 8C-1. Differences Between State And Overview Measurements

OVERVIEW ON A RANGING LABEL

This type of measurement records the occurrence of overview events found on a ranging label during program flow. Each overview event represents one of the states or ranges of states of interest. The analyzer classifies each state and stores the corresponding event number in memory. After the overview measurement is complete, you can obtain displays of information about the overview events. Figure 8C-2 is a block diagram showing the way the analyzer makes an overview measurement of states on a ranging label.

Setup: Prior to making this measurement, you designate the ranging label, identifying the set of incoming lines from the ranging pod to be monitored (e.g. (overview) (on) (ADDRESS)). You also define the event specifications which will identify the states or ranges of states found during the measurement (e.g. (overview) (event) 1 (is range) 1000H (thru) 2000H). Make both of these entries in the trace specification.

Execute: When the analyzer makes the measurement, it monitors the lines identified by the ranging label you specified, and sends the states from those lines to the event detector.

The event detector compares each state that arrives with the event specifications that were set up prior to the overview measurement. If it finds that a state meets any of the event specifications, it sends the 4-bit code for that event to the overview memory. If a state does not meet any of the event specifications, the "everything else" code is sent to the overview memory. The "everything else" code is not stored unless you specified its storage before the start of the measurement.

You can qualify your label to require that the overview measurement be made when certain other conditions are true if you overview-enable using the sequence. In this case, overview enable is a current-state enable.

You can use the sequence or one of the window elements to window the overview measurement. Set up the sequence or window element to enable only during the portion of program execution that you want to examine. Then set up the overview memory to store events only during the enable period of that element.

The event detector also sends the event codes to the trigger circuit for trigger generation on overview events, if specified. Triggers can only be generated when a trigger-specified event is actually stored in overview memory (during enable periods, if you are windowing your measurement). You cannot use the trigger enable function to additionally window overview triggers. All trigger windowing on an overview event is included within the overview enable specification.

The overview memory accepts the event codes and stores them in the order they occur.

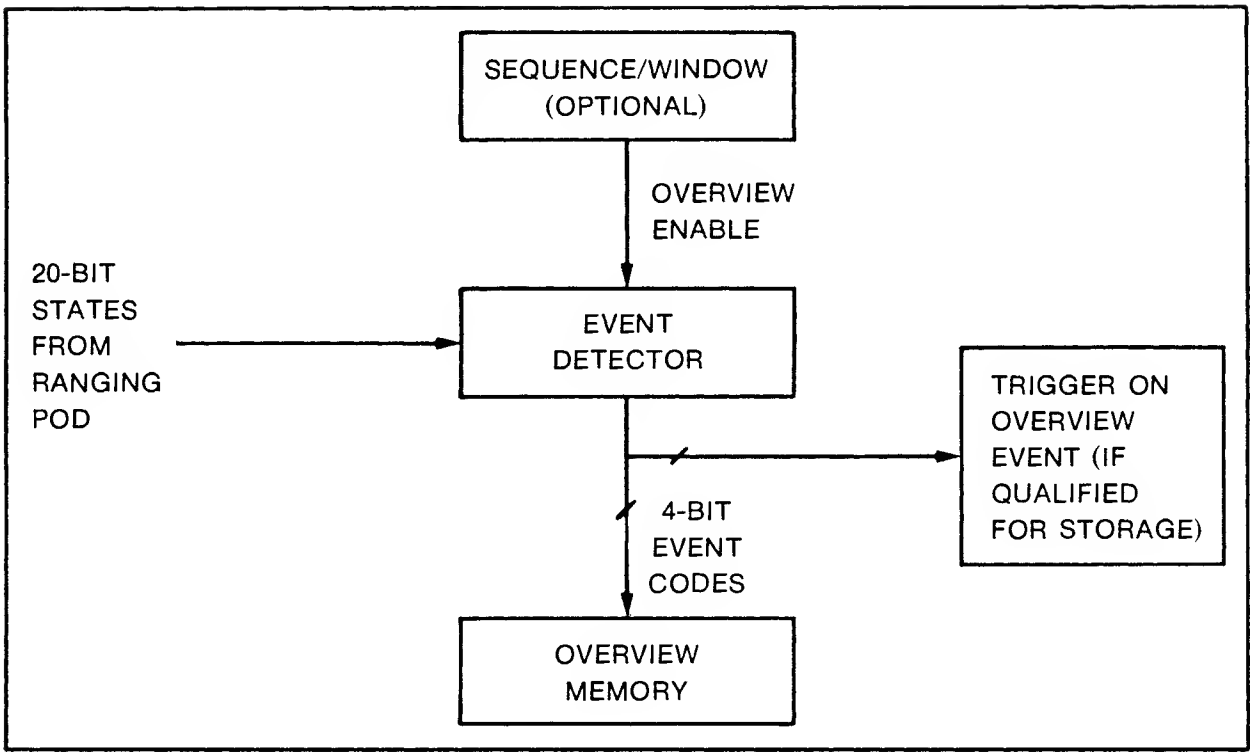


Figure 8C-2. Overview On A Ranging Label

OVERVIEW ON STATE COUNT

This overview measurement records the numbers of states that occur between some start point and stop point in program flow, usually the beginning and ending of a routine of interest. Each overview event represents some count or range of counts, from 0 to 720 billion counts. The display shows how often the count of states fell within the specifications of each of the events. Figure 8C-3 is a block diagram showing the elements used to make an overview measurement of state counts. Figure 8C-4 shows the way the analyzer makes an overview measurement on state count.

Setup: Prior to starting the measurement, you set up the sequence or one of the windows with the states where you want your counts to start and stop. When the output from the sequence or window switches to enable, the overview function will reset its counter and start the measurement. When it switches to disable, the overview function will stop its counter and store the corresponding overview event. Now you enter the event specifications, selecting events to represent different counts or ranges of counts.

Example entries for Overview on State Count:

```
(overview) (_on_) (state_cnt)

(window) (_one_) (enable) (ADDRESS) (__=__) 1C8H (disable)
                  (ADDRESS) (__=__) 29EH

(overview) (enable) (window) (_one_)

(overview) (event)  1 (is_range)  1 (counts) (thru)   100
                                               (counts)

(overview) (event)  2 (is_range)  101 (counts) (thru)   200
                                               (counts)

(overview) (event)  3 (is_range)  201 (counts) (thru)   300
                                               (counts)
```

Execute: The sequence or window element that is used by the overview measurement begins in the disable condition. Each state from all pods is compared with the enable specification. When the sequence or window finds the state that satisfies its enable specification (window one finds 1C8H in the example), it enables. The enable transition resets the state counter to zero. Now as each new state arrives, the counter will increment its count by one. In the meantime, the sequence or window element looks for the state or states that satisfy its disable specification.

The count from the counter is sent to the event detector where it is classified. The event detector outputs a 4-bit code which represents the event whose specification is met by the present count.

When the sequence or window element finds the state that satisfies its disable specification (window one finds 29EH in the example), it switches its output to disable. The disable transition causes the overview memory to store the present event classification (4-bit code), and allows the trigger circuit to recognize trigger if the present event was also specified as an overview trigger. You cannot additionally window your overview triggers using trigger enable. All windowing of overview triggers must be part of the overview specification.

The sequence or window element now begins its search for the state or states that will enable it. When its enable specification is satisfied, it will again reset the counter to zero and start a new count of states.

The states will be counted between each beginning and ending point until either the overview event memory is filled with event codes or you halt the measurement.

The following characteristics affect overview measurements of state counts:

a. The sequence or window that is used to control the state count must enable and disable before the first overview event is measured in order to ensure that the counter is reset.

b. The state that enables the sequence or window is never included in the count.

c. The state that disables the sequence or window is never included in the count.

d. The state that immediately precedes the end of the count (last state before disabling state) may or may not be counted, depending on the sampling clock rate. The state will not be counted when the clock rate is above approximately 7 MHz, even if it is otherwise enabled and qualified.

You can include another sequence or window element in your overview specification to obtain a second level of windowing in your measurement. For example, you might want to overview interrupt processing, but analyze only the usage of a masking subroutine. By enabling your count interval during the interrupt routine and then qualifying your count on the addresses of the masking subroutine, you can obtain events which correspond to the usage of the mask subroutine for each call of the interrupt handler. Example, (overview) (enable) (window) (one) and (overview) (count) (enable) (window) (two).

By setting up the state/software analyzer to count occurrences of a specific state, you can overview occurrences of transactions of interest during the routine of the measurement. Example, (overview) (count) (ADDRESS) (==) INTERRUPT (map) (ADDR_MAP).

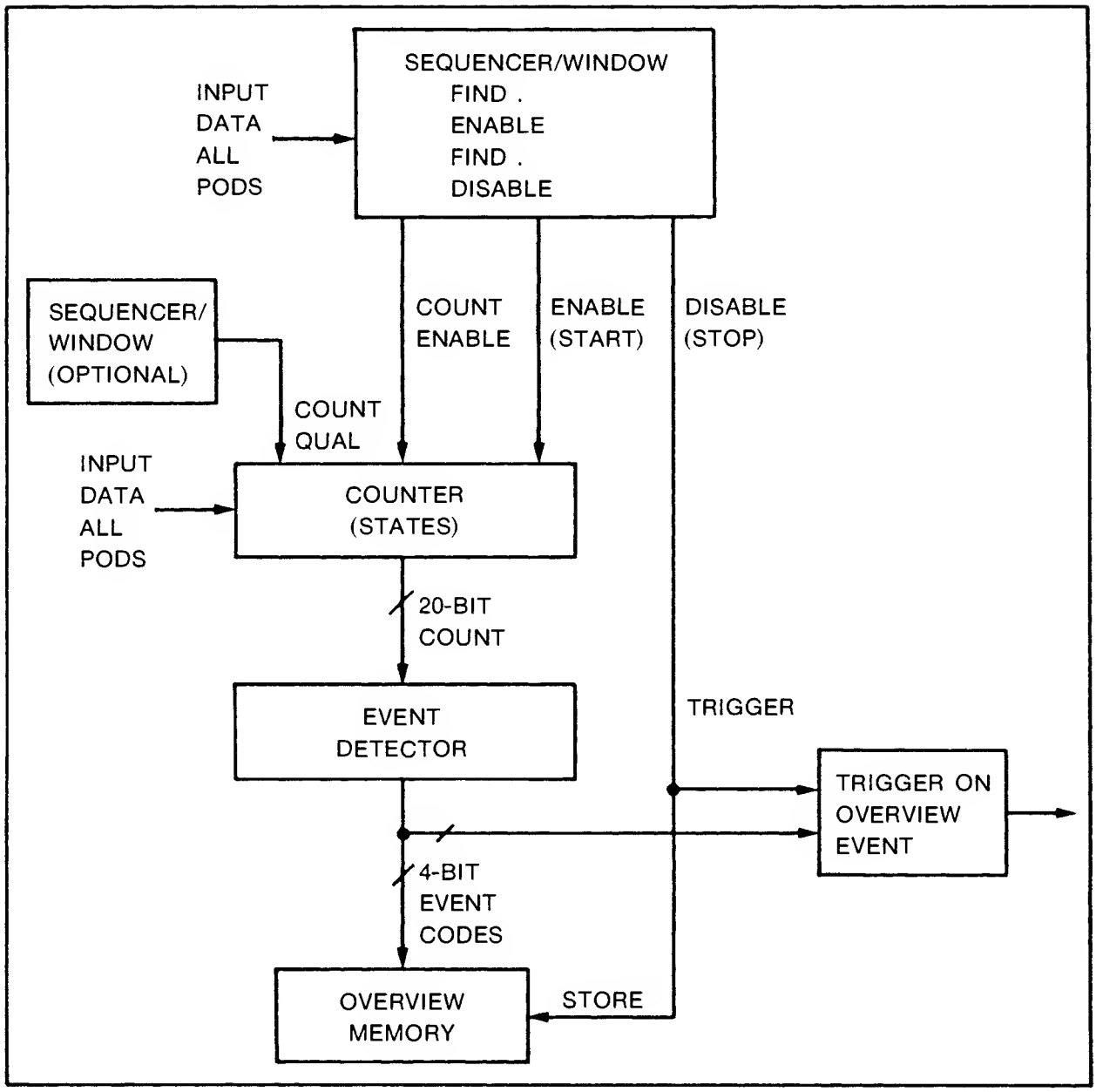


Figure 8C-3. Overview On State Count

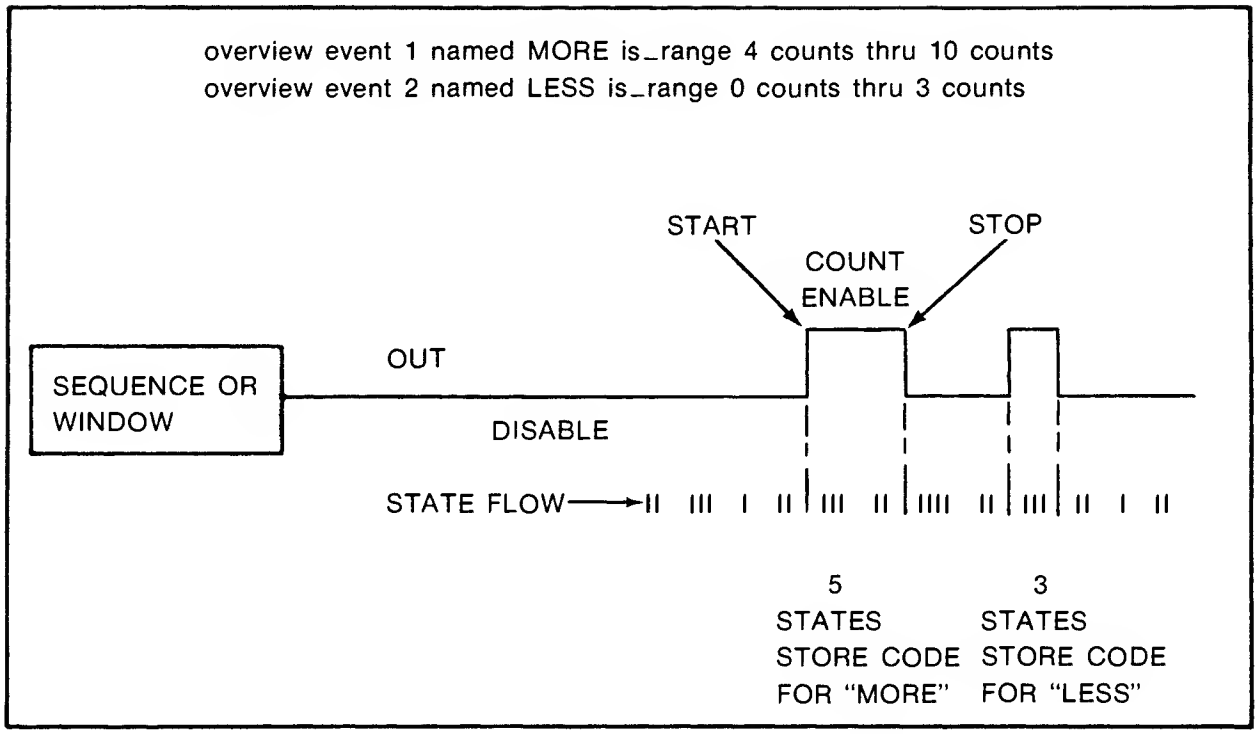


Figure 8C-4. Example Measurement Of Overview On State Count

OVERVIEW ON TIME COUNT

Overviews of time counts measure periods of time that elapse between selected start points and stop points in program flow (defined by a window). Each overview event represents some range of time periods, from 0 to 480 minutes. The overview display shows how often the time measurements fell within the specifications of each of the event ranges. Figure 8C-5 is a block diagram showing the analyzer elements used to make an overview measurement of time.

Setup: Before starting the measurement, set up the sequence or one of the windows with the state which you want to enable and the state which you want to disable each time measurement. The period of the enable output from the sequence or window will be the time measured. Now you enter the event specifications. Set up each event to represent a different range of time periods.

Example entries for Overview on Time Count:

```
(overview) (_on_) (time_cnt)

(window) (_one_) (enable) (ADDRESS) (___=___) 1C8H (disable)

                  (ADDRESS) (___=___)  29EH

(overview) (enable) (window) (_one_)

(overview) (event) 1 (is_range)   0 (usec) (thru)

                                 10 (usec)

(overview) (event) 2 (is_range)  10 (usec) (thru)

                   20 (usec)

(overview) (event) 3 (is_range)  20 (usec) (thru)

                   30 (usec)

(overview) (event) 4 (is_range)  30 (usec) (thru)

                   40 (usec)
```

Execute: Before measuring the first overview time interval, the sequence or window that is used to control the time measurement must enable and disable in order to ensure that the counter is reset. The overview begins with the sequence or window element in the disable state. Each state is compared with the enable specification. When the sequence or window finds the state that satisfies its enable specification (window one finds 1C8H in the example), it changes its output to the enable level, resetting the timer to zero. This starts a new measurement of time. Now the sequence or window element looks for the state or states that satisfy its disable specification.

The measure of time from the counter is sent to the event detector where its event classification is determined. The event detector outputs an event code corresponding to the present event (range of time) you set up before the run. (It outputs the code for event 1 while the time is between 0 and 10 usec in the example.)

When the sequence or window element finds the state that satisfies its disable specification (window finds 29EH in the example), it switches its output to disable. The disable transition causes the overview memory to store the present event code from the event detector. The transition to disable also allows the trigger circuit to recognize trigger if the stored event was also specified as an overview trigger. You cannot window overview triggers using trigger enable. All windowing of overview triggers must be included in the overview specification.

The sequence or window element now returns to its search for the state or states that will enable it. When its enable specification is satisfied, it will again reset the counter to zero and start a new measurement of time.

A new measurement of time will be made between each occurrence of your start and stop specifications until either the overview memory is filled with event codes or you halt the run, depending on the mode of measurement.

You can include another sequence or window element in your overview specification to obtain a second level of windowing in your measurement, just as was explained for the overview of state counts. If you were overviewing the time required to execute an interrupt processing routine, but you only wanted to overview that portion that involved a masking subroutine, you could enable your count inverval during the interrupt routine, and then qualify the count interval during the address range of the mask subroutine. In this way, you could obtain events which correspond to the execution period of the mask subroutine during calls of the interrupt handler.

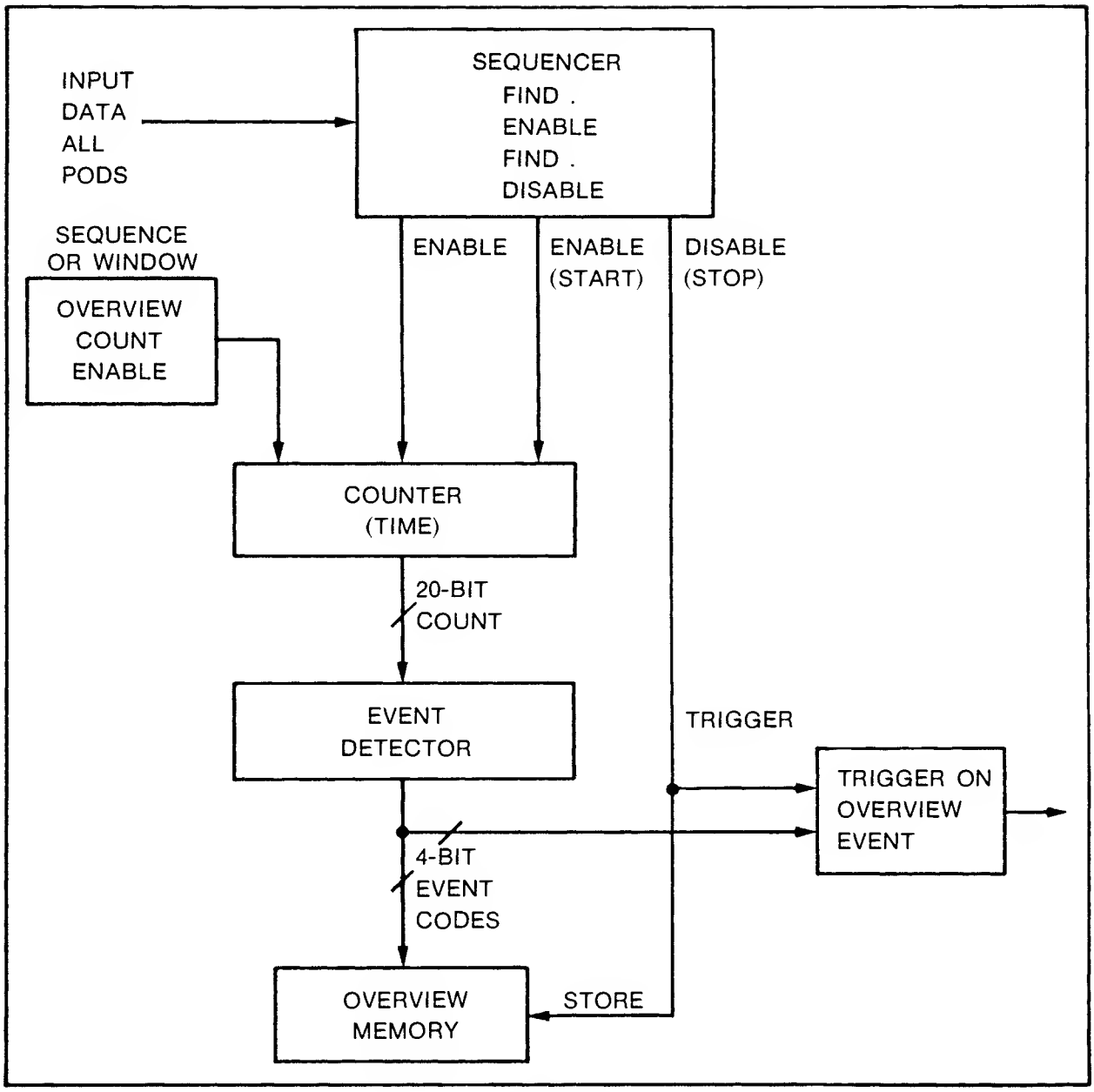


Figure 8C-5. Overview On Time Count

SPECIFYING OVERVIEW EVENTS

When you set up the state/software analyzer to make an overview measurement, you divide the range of activity to be included in the measurement into events (ranges and/or points). This is true whether you are dividing ranges of states, ranges of state counts, or ranges of time periods. Then you assign event numbers and names to identify each of these ranges. The names will appear on screen when the analyzer displays the results of the overview measurement.

If you are dividing a range of addresses to make an overview measurement of state flow, you might assign overview events to the ROM, RAM, DISPLAY, STACK, and I/O addresses. During the run, the state/software analyzer will accumulate separate counts of the number of states it finds in each of these overview events. After the trace, you will be able to see the number of program transactions that were made within each of these overview events (address ranges). Figure 8C-6 shows how you might divide an address range into events, and how you might choose names for each event. You cannot overlap your event specifications because the analyzer cannot store two event codes to identify the same event. Also, the upper bound of each event range must be equal to or greater than the lower bound.

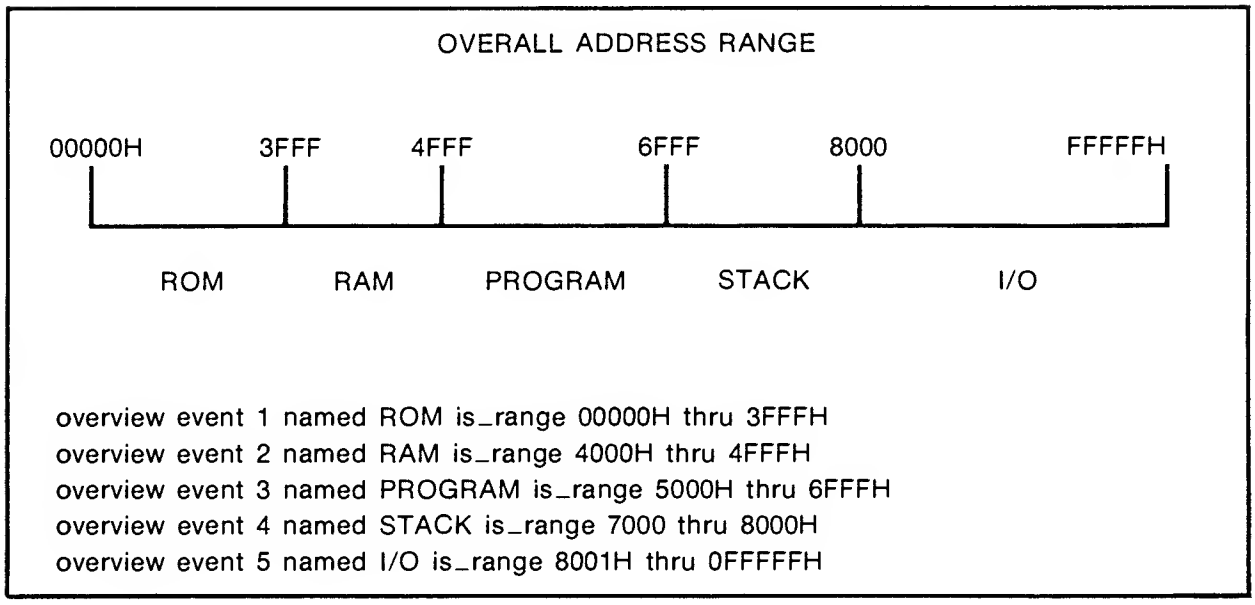


Figure 8C-6. Choosing Named Events

All of the symbol capabilities available for trigger, store, and count specifications are also available for specifying overview events. Using figure 8C-6 as an example:

a. (overview) (event) 4 (named) STACK (is_range) (STACK) will enter the event if the symbol STACK is defined on the default map for the ADDRESS label.

b. (overview) (event) 4 (named) STACK (is_range) STACK (map) (ADDR_MAP) will enter the event if the STACK symbol is defined on the ADDR_MAP (not the default map assigned to the ADDRESS label).

c. (overview) (event) 4 (named) STACK (is_range) (PROGRAM) + 1 (thru) (I_O) - 1 will enter a specification for STACK if it is not defined on the default map, but PROGRAM and I_O are defined there.

d. (overview) (event) 6 (named) UPDATE (is_range) SCRATCH_PAD (data) (start) + 50 (thru) KEYBOARD (prog) (end) - 7. This specification for event 6 is an example using symbols defined in the absolute file in the development system. You can also define ranges of overview events using source code line numbers.

e. (overview) (event) 12 (is_range) 1000H (thru) 2000H. This specification is an example of entering an event without assigning a name. In this case, your overview displays will show this as the event named '1000H thru 2000H'.

DETECTING THE OVERVIEW EVENT

The overview event detector receives 20-bit input states. These come from the ranging label when you are making an overview measurement of state ranges. Otherwise, these come from an internal floating point counter in the analyzer. If you are making a state-count overview, the 20-bit input is the current count of states. If you are making a time-count overview, the 20-bit input is the elapsed time since the last "start-count" transition.

The 20-bit input is compared with the event specifications in the event detector. If it falls within one of the event ranges or is one of the event points, the overview event detector outputs the 4-bit code corresponding to that event. Figure 8C-7 shows the way the analyzer detects overview events. Values are ignored if no overview event was assigned to represent them.

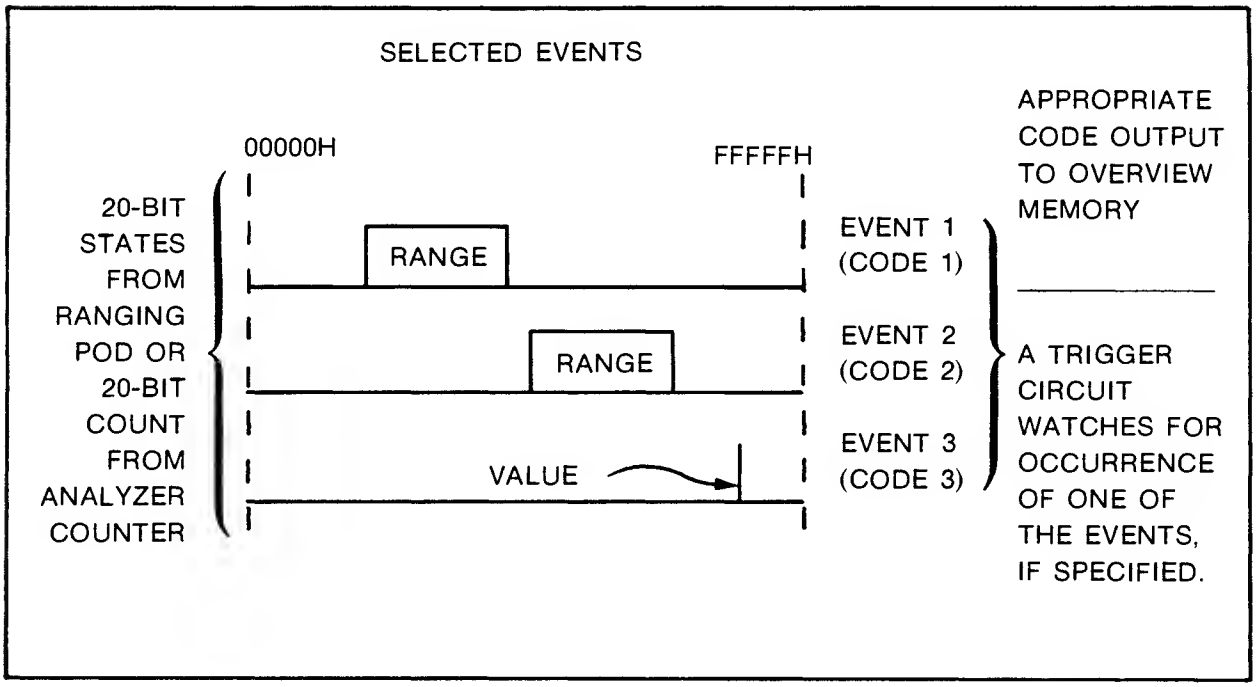


Figure 8C-7. Detecting The Overview Event

Each time a 20-bit state arrives from the ranging data pod or from the internal counter, it is compared with the event specifications you set up before the measurement. If the 20-bit number matches one of the events (falls within one of the specified ranges, or is a state that is one of the specified points), then the overview event detector will produce the code for that event. If the 20-bit number does not match any of the overview events, then it is classified as an "everything else" event. No event code is stored during the corresponding time period unless you specified storage of the "everything else" events.

During the trace, the trigger circuit watches the flow of event codes from the event detector. It will generate a trigger if it stores a particular event that you specified as a trigger event. This trigger generation can be used to capture a block of state flow around a particular occurrence, such as when a routine takes too long a time (or too short) to execute. Simply assign an event name and number to the range of abnormal time. Then set up the analyzer to trigger a trace if that event is ever stored during your measurement. Now you can capture the state-by-state transactions associated with that abnormal event and examine them in the trace list.

THE OVERVIEW UNTIL SELECTION

You can have the analyzer make a single overview measurement or a continuous overview measurement. The (overview) (until) selection allows you to choose between these two operating modes.

(overview) (until) (user_halt) selects the continuous mode of measurement. Overview events will be continuously stored in memory until you press the (halt) softkey and (RETURN).

(overview) (until) (mem_full) selects the single measurement mode. The overview measurement stores overview event codes in memory until 4096 event codes have been stored. Then the analyzer automatically completes the measurement and displays its overview information.

EXAMPLE: SELECT ROUTINES TO OPTIMIZE

Assume that there are three complex routines in your target program and you think that they might be taking a lot of program time. You would like to see how much of the total program time is taken by each of the routines.

Set up an overview measurement with named events for the address ranges of the three routines, as follows:

```
(overview) (event) 1 (named) PGM_1 (is_range) <address range of
                                               first routine>
(overview) (event) 2 (named) PGM_2 (is_range) <address range of
                                               second routine>
(overview) (event) 3 (named) PGM_3 (is_range) <address range of
                                               third routine>
```

You can use labels in place of numeric values, if you set up the labels on a map in the Format Specification.

Now execute an overview measurement until user halt and call up display of a histogram. The "until user halt" mode causes the overview memory to be continuously updated as the program runs. Your histogram display might be as shown in figure 8C-8.

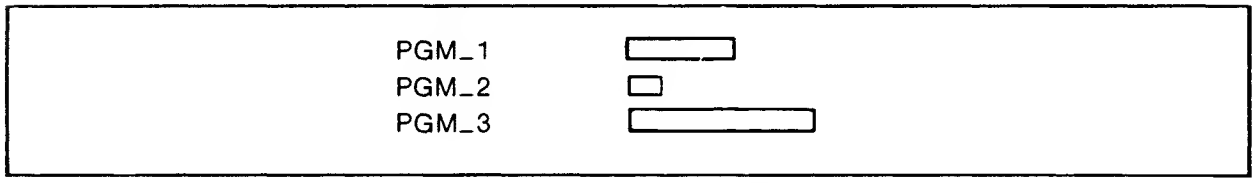


Figure 8C-8. Analyzing Subroutine Run Times

The display in figure 8C-8 indicates that you will gain the most advantage by optimizing program 3, and then program 1. There does not seem to be much system time spent in program 2.

Before you go ahead and work on program 3, you should add one more component to your histogram display: the "everything else" event. Now you might get a display like figure 8C-9.

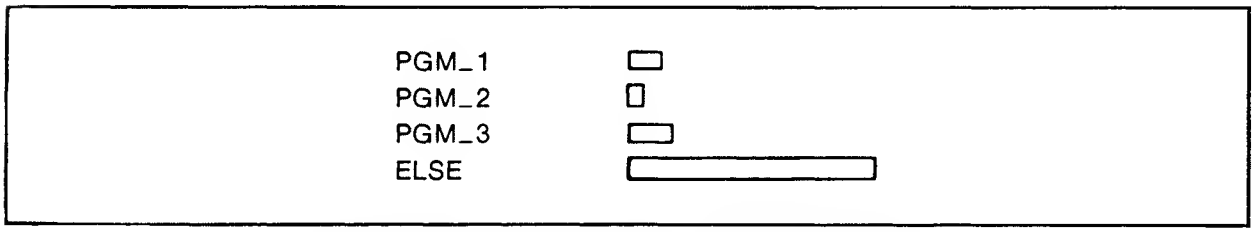


Figure 8C-9. Adding The "Everything Else" Event

From figure 8C-9, you can see that most of the program time is spent executing activity in other areas of the address space. The "everything else" event is very handy at a time like this because it can keep you from wasting time optimizing the wrong routines. Now you can divide the "everything else" event into its component subroutines and find out which program routines are really taking the greatest amount of time.

EXAMPLE: TRIGGERING A TRACE ON AN OVERVIEW EVENT

You might be using an overview measurement to characterize an interrupt handler routine. You will have set up a sequence or window element with the starting and ending addresses of the interrupt handler routine, and you will use it to enable your overview measurement. Then you will have assigned event numbers and names to identify each of the address ranges used in normal interrupt processing. You can use a histogram display to observe the usage of each of the subroutines involved in normal interrupt processing. See figure 8C-10 for the example routine.

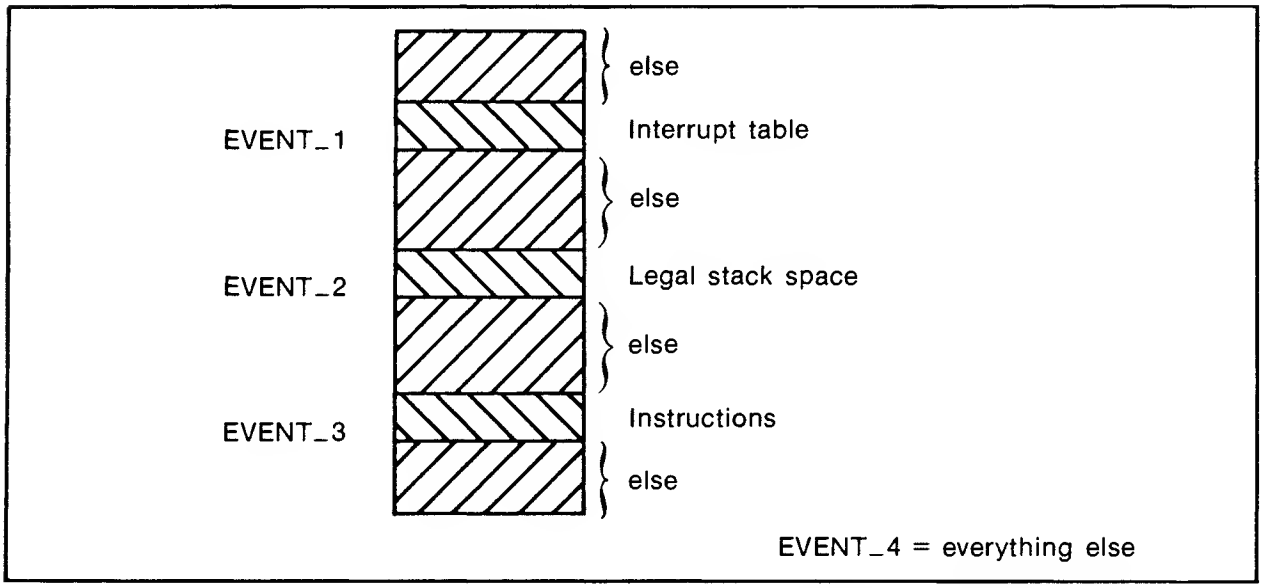


Figure 8C-10. Example Interrupt Address Space

While your interrupt handler routine is in use, you might also like to know if any of the illegal address space is ever accessed. If so, you would like to examine the state-by-state transactions that led up to the access of that illegal address space. You can assign an additional overview event to all of the remaining address blocks that together make up the illegal address space. Then specify triggering a trace if the "everything else" event is ever stored (trigger on overview 4).

EXAMPLE: EXAMINING EXECUTION TIMES OF ROUTINES

You might like to know if the execution time of a routine is stable. This is easy to check with an overview measurement. Set up a window to identify the beginning and ending addresses of the routine in question. You can use a label to identify the address range of the routine if you set up that label on a map in the Format Specification.

```
(window) (_one_) (enable) (ADDRESS) (__=__) <label of entry point>

                 (disable) (ADDRESS) (__=__) <label of exit point>
```

Then set up the analyzer to make an overview measurement during the period when window one provides the enable output.

```
(overview) (enable) (window) (_one_)

(overview) (_on_) (time_cnt)
```

Now set up the overview events to cover the anticipated range of execution times for your routine. Perhaps two events will be sufficient: one for the normal range of execution times, and the other for the abnormal range of execution times.

```
(overview) (event) 1 (is_range) 10 (usec) (thru) 20 (usec)

(overview) (event) 2 (is_range) (else)
```

You can set up your analyzer to trigger if the execution time of your routine is ever outside the 10-usec to 20-usec normal execution time.

```
(trigger) (_on_) (overview) 2
```

With this setup, the analyzer will measure the execution time between the start and end of your subroutine everytime the subroutine is executed. Then if the execution time of that subroutine is ever outside of the normal range (event_1), the analyzer will trigger a trace of states. You should select a trigger position that is near the end of trace memory so that you can store the transactions before and after the abnormal routine execution.

EXAMPLE: CHECKING RETRIES OF DISC READ IN TARGET SYSTEM

You might suspect that the target system occasionally has to retry reads of disc memory an abnormal number of times. The overview measurement system can help you isolate the software associated with the abnormal retrys. Set up the analyzer trace specification as follows:

```
(window) (_one_) (enable) (ADDRESS) (__=__)<start of disc-try
routine> (disable) (ADDRESS) (__=__) <recognition
of successful disc-try>
```

Now set up the analyzer to do a state-count overview. Instead of counting all states that are executed, just count the states that signify retries. Perhaps two events will be sufficient.

```
(overview) (_on_) (state_cnt)

(overview) (enable) (window) (_one_)

(overview) (count) (_on_) (ADDRESS) (__=__) (RETRY)

(overview) (event) 1 (is_range) 0 (thru) <maximum acceptable
                                          number of times for
                                          retrying disc>

(overview) (event) 2 (is_range) (else)
```

Now set up the analyzer to trigger a trace if the target system ever executes an abnormal number of states.

```
(trigger) (_on_) (overview) 2
```

With this setup, the analyzer will collect a continuous stream of states into its trace memory until either you halt the run or the target system executes an abnormal number of disc retries. Select a trigger position deep within the trace memory to allow examination of the states leading up to the abnormal condition.

Chapter 8D
INTERMODULE MEASUREMENTS

INTRODUCTION

Intermodule measurements are measurements involving coordination between two or more analysis modules. This coordination uses high-speed communication between all analysis modules. It coordinates triggering, windowing of functions, and synchronization of execute and halt for all modules involved in a measurement.

This chapter describes the intermodule measurement capabilities of the state/software analyzer. It describes the way the state/software analyzer uses the lines of the intermodule bus to interact with other analyzers installed in the mainframe. It lists and defines the state/software analyzer command syntax for setting up intermodule measurements. Finally, two example measurements are described: one combining two state/software analyzers, and the other using a state/software analyzer and timing analyzer together. Refer to the Measurement System Reference Manual for information about overall measurement system interaction.

GENERAL INFORMATION

Interactive measurements are made through the intermodule bus cable which is connected to the IMB connector on the control cards of all analyzers. The purpose of the intermodule bus is to synchronize executing and halting measurements which involve two or more analyzers.

You can have the state/software analyzer receive enable levels or triggers from other analyzers in the system. You can also have the state/software analyzer drive these enable levels and triggers to the other analysis modules on the intermodule bus. Additionally, the state/software analyzer can provide a delay clock signal to be used by other analysis modules on the intermodule bus.

At power-up, no specification is made for any interactive measurement between the state/software analyzer(s) and other analysis modules. In order to obtain interactive measurements, select the state/software analyzer trace specification and enter the interactions desired. (The softkeys for making these entries are all in the trace specification. They only appear on screen in instrument configurations which have two or more analysis modules installed.)

INTERMODULE SIGNALS

There are five control signals on the intermodule bus. The state/software analyzer can participate in activity on all five. The

signals are described in the followng paragraphs:

MASTER ENABLE

Any analyzer involved in an intermodule measurement will automatically receive the master enable line to synchronize its execute and halt functions. The master enable line carries a logic level which is either true or false. When it is true, it enables all instruments that receive it. When it is false, it disables state/software analyzer measurement functions (including the sequence elements and overview measurements).

NOTE

When the state/software analyzer is in the "internal_and_driven" mode, the master enable function will disable all but the sequence or window element it is using during disable periods. If you select "driven_only" mode, the master enable function will only disable the other analysis modules during disable periods.

There can be only one driver for the master enable line in the state/software analyzer: either the (execute)/(halt) function (the default configuration), or one of the analyzers assigned to participate in the measurement.

Measurements can be executed and halted from any analyzer in the mainframe or from the measurement system display, no matter which unit is assigned to control master enable.

The purpose of the master enable line is to synchronize measurement start in all analyzers. At measurement start ((execute) key pressed or true state from designated master enable driver), each analyzer tries to start. The master enable line delays start until all analyzers are ready. Then the master enable line switches true, releasing all analyzers to start together.

When the master enable line is driven by one of the analyzers, it can alternate between true and false during a measurement to window the activity included in the interactive measurement. When the state/software analyzer receives this line from some other analysis module, all of its analysis functions are disabled during disable periods.

MASTER ENABLE COMMAND SYNTAX

a. (master) (enable) (received). All measurement functions of the state/software analyzer are disabled until it receives master enable from some other analyzer on the intermodule bus. This is the default setup when any other analyzer is designated to drive the master enable function (synchronize the measurement).

b. [master] [enable] [driven] [sequence]. This is the "driven_only" mode. Functions within the state/software analyzer are not controlled by master enable. When the state/software analyzer recognizes its sequence-enable specification, it drives the master enable level to the other instruments on the intermodule bus. The [sequence] can be replaced by one of the [window] selections, if desired.

c. [master] [enable] [int/drive] [sequence]. This is the "internal_and_driven" mode. Functions within the state/software analyzer are controlled by master enable. All measurement functions in the state/software analyzer remain disabled (except for the sequence element used by master enable) until the state/software analyzer recognizes its sequence-enable specification. Then master enables the analysis functions and drives the enable level onto the master line of the intermodule bus. The [sequence] can be replaced by one of the [window] selections, if desired.

TRIGGER ENABLE

The trigger enable line carries a logic level. When it is true, it enables the receiving instruments to recognize their internal triggers, if they occur. When it is false, it disables trigger recognition in the receiving instruments. The trigger enable line can alternate between true and false during a measurement to allow the controlling analyzer to window the program activity where trigger recognition can occur. The state/software analyzer can drive this line, or the state/software analyzer can receive this signal from some other instrument on the intermodule bus. If no analyzer is designated to drive this line, it will default to the true state.

TRIGGER ENABLE COMMAND SYNTAX

a. [trigger] [enable] [received]. The state/software analyzer cannot search for its internal trigger until it receives its trigger enable from some other analyzer on the intermodule bus. If [trigger] is also received, this selection will have no effect.

b. [trigger] [enable] [driven] [sequence]. The state/software analyzer is always able to search for its own trigger, but the other instruments receiving the enable level are not. When the state/software analyzer satisfies its sequence specification, it places the enable level on the intermodule bus, allowing the other instruments to search for their triggers. A window can be substituted for the sequence, if desired.

c. [trigger] [enable] [int/drive] [sequence]. The state/software analyzer cannot search for its trigger until it satisfies its own sequence specification. Then it drives the enable line true on the intermodule bus, and begins its search for a trigger. A window can be substituted for the sequence, if desired.

d. [trigger] [drive_te] [on] [ADDRESS] [=] [DISPLAY]. This mode of triggering can be specified to occur on a wide variety of conditions. The state/software analyzer will recognize the occurrence you specify as

the trigger event. When it recognizes its trigger event, the state/software analyzer will drive the trigger-enable line true in the intermodule bus.

TRIGGER

The trigger line carries a transition from false to true. When the trigger line switches from false to true during a measurement, it remains true for the rest of the measurement. The trigger transition can be driven by the state/software analyzer when it recognizes its trigger specification. The state/software analyzer can also receive the trigger transition from another analyzer on the intermodule bus. In this case, the state/software analyzer will identify the state at its input as the trigger state in its trace memory and on the trace list display. The trigger line can have more than one driver designated. When more than one driver is designated, the first driver to recognize its trigger and issue the trigger transition becomes the trigger source for the measurement in process (this is the "received_or_driven" mode).

TRIGGER COMMAND SYNTAX

a. (trigger) (received). The state/software analyzer recognizes trigger when it receives the appropriate transition from the intermodule bus. At this case, any state at the analyzer input will be identified as the trigger state in the trace memory. The trigger enable specification will have no effect when operating in this mode.

b. (trigger) (int/drive) (on) (ADDRESS) (≡) (DISPLAY). This mode of triggering can be specified to occur on a wide variety of conditions. When the state/software analyzer recognizes its own internal trigger specification, it drives the trigger transition on the trigger line of the intermodule bus.

c. (trigger) (rec/drive) (on) (ADDRESS) (≡) (DISPLAY). This mode of triggering can be specified to occur on a wide variety of conditions. This command line is used to involve the state/software analyzer in an OR'd trigger configuration. The state/software analyzer will recognize its trigger either when it receives the transition from the trigger line of the intermodule bus, or when it satisfies its own internal specification. If it satisfies its own trigger specification first, it will drive the trigger transition onto the intermodule bus. If you specify a received trigger enable, it will affect the internally generated trigger.

STORAGE ENABLE

The storage enable line carries a logic level. A state/software analyzer can drive this line, controlling when the receiving instruments are allowed to store information. A state/software analyzer can receive this line from another analyzer; it will store qualified states in trace memory when this line is true. The storage enable line has no effect on an overview measurement. The storage enable line can alternate between true and false during a measurement to window activity which can be

stored. If no analyzer is designated to drive this line, it will default to the true state.

STORE ENABLE COMMAND SYNTAX

a. [store] [enable] [received]. The state/software analyzer receives its store enable level from some other analyzer on the intermodule bus. It will store states anytime this enable is true.

b. [store] [enable] [driven] [sequence]. The state/software analyzer drives the enable level on the intermodule bus when it recognizes its internal specification. One of the window elements can be specified instead of the sequence, if desired.

c. [store] [enable] [int/drive] [sequence]. The state/software analyzer follows the sequence output. When it is enabled, the state/software analyzer enables its store function and drives the enable level on the appropriate line of the intermodule bus. When sequence is disabled, the state/software analyzer disables its store function, and the store functions of all receiving analyzers. One of the window elements can be specified instead of the sequence, if desired.

DELAY CLOCK

The delay clock line carries a series of transitions. The state/software analyzer can issue these clocks each time it recognizes its incoming clocks. The state/software analyzer cannot receive clocks from this line. Other analyzers (such as timing analyzers) can receive these clocks and use them to count delays when taking their measurements (delay by number of clocks).

DELAY CLOCK COMMAND SYNTAX

[master] [delay_clk] [driven]. When the state/software analyzer recognizes its own external clock input, it issues clocks to other analyzers on the intermodule bus. If not in use, select [off].

EXAMPLE: USING ONE STATE/SOFTWARE ANALYZER TO TRIGGER ANOTHER STATE/SOFTWARE ANALYZER

The following paragraphs describe how to set up two state/software analyzers to make a measurement involving interactive triggering:

a. From the monitor level of softkeys, press (meas_sys) and (RETURN). This will gain access to the measurement system display. Add (continue) to your command if you want the state/software analyzers to continue using their former test setups. The softkey label line will identify both state/software analyzers by name and by the slot numbers where their control cards are installed.

b. Your identifiers will be similar to this example line:

state_0 state_4 _____ _____ _____ end

c. Press one of the state softkeys (such as (state_0)) to select the state/software analyzer you want to set up first. Add a file name if you want to load a configuration from file memory. Then press (RETURN). The trace specification of the state/software analyzer whose control card is installed in the selected mainframe slot (slot 0 in this example) will appear on screen.

d. Make any required modifications to the specifications of the selected analyzer.

e. Set up this state/software analyzer to drive the trigger transition on the intermodule bus. Press the following softkeys in the trace specification:

(trigger) (===ETC===) (int/drive) (_on_) (sequence) (enable)

(RETURN)

f. Press the (end) softkey and (RETURN). The CRT will again show the measurement system display and will identify the interaction you assigned to your state/software analyzer (trigger Driver(s) State_0).

g. Press the other state softkey (such as (state_4)). Again, decide whether you want to use the default test setup or load a configuration from file memory (by adding the name of the file). Press (RETURN). This activates the other analysis module and brings its trace specification to the screen.

h. Make any desired changes to the measurement setup for this state/software analyzer.

i. Set up this state/software analyzer to receive its trigger from the intermodule bus. Press the following softkeys in the trace specification:

(trigger) (===ETC===) (received) (RETURN)

j. Press (end) and (RETURN) to return to the measurement system level of display. This display will show the interactions you specified for both state/software analyzers (trigger Driver(s) state_0, Receiver(s) state_4).

NOTE

You can execute or halt interactive measurements while the system displays the measurement system monitor or while it displays specifications of any of the analyzers involved in the measurement.

k. Press (execute) and (RETURN). This starts both state/software analyzers running according to their own measurement configurations. The analyzer called state_0 will begin looking for its trigger. The other analyzer will wait to receive its trigger pulse from the intermodule bus. When the sequence element in state_0 switches to the enable state, state_0 will accept this as its trigger. Then it will send the trigger transition on the intermodule bus to state_4. When the analyzer called state_4 receives the trigger transition, it will recognize the state present at its input as its trigger state.

NOTE

During the measurement, you can select any of the other analyzers in your mainframe and look at their specifications, but you cannot execute measurements with them.

l. Press the (end) softkey again. This returns you to the monitor level of display. You can use the mainframe for nonanalysis activity (edit, assembly, etc). Both state/software analyzers will continue to execute their measurements according to their specifications and save data in their memories.

m. When you want to look at the results of the measurements, press the (meas_sys) and (continue) softkeys. Then press (RETURN). This command line returns the system to the measurement system display without changing the analyzer setups.

n. Now press (state_<number>) and (RETURN). This regains access to the analyzer you select without reloading the module. The measurement you assigned to this analyzer will still be running, if it was not yet completed.

o. When you have finished with that analyzer, press (end) and (RETURN).

p. You can check the status of the measurement in the other analyzer. Press (state<its number>) and (RETURN).

EXAMPLE: INTERMODULE MEASUREMENT USING A STATE/SOFTWARE ANALYZER AND TIMING ANALYZER TOGETHER

Assume there is a system that occasionally exits normal software while it is in an I/O driver routine. You think that there may be a hardware problem in one of the control lines, such as a short-duration pulse (glitch), that may be causing the error condition.

You can set up the timing analyzer to trigger if a glitch occurs on any of the control lines. Then you can set up the state/software analyzer to enable timing analyzer trigger recognition only when the software is executing the I/O driver routine.

a. In the measurement system monitor, select the timing analyzer and set it up to trigger if a glitch occurs on any one of the suspected control lines. Then enter the commands to receive trigger-enable from the intermodule bus.

b. Press the (end) softkey and (RETURN). This returns the system to the measurement system monitor.

c. Press the (state_<number>) softkey, and (RETURN).

d. Set up window one to enable at the start of the I/O driver and disable at the end of the I/O driver. Then press the following keys:

(trigger) (enable) (===ETC===) (driven) (window) (one) (RETURN)

e. Press the (execute) softkey, and (RETURN).

f. This starts the measurement system. The state/software analyzer will enable the timing analyzer to search for its trigger each time the system under test executes the I/O driver routines. If the timing analyzer finds a glitch on any of the control lines during the I/O driver routines, it will trigger a timing measurement, capturing the glitch and the related activity. If the timing analyzer finds a glitch on any of the control lines at any other time, it will not recognize it as a trigger.

Assume you have found a glitch occurring on one of the control lines. You would like to know if the glitch is caused by some software problem in the I/O driver routine. You can add a specification to the above setup so that the timing analyzer will trigger a trace in the state/software analyzer when it detects the glitch.

a. Return to the timing analyzer from the measurement system display and enter the command to make the timing analyzer drive the trigger transition on the intermodule bus. Do not disturb any other part of the timing analyzer specification.

b. Press the (end) softkey, and (RETURN). This returns the system to the measurement system display.

c. Press the (state_<number>) softkey, and (RETURN).

d. Press the following softkeys in the trace specification:

(trigger) (===ETC===) (received) (RETURN)

e. This sets up the state/software analyzer to recognize its trigger when the timing analyzer detects its glitch. Select a trigger position that will allow you to examine state activity before and after the error occurrence.

f. Press the (execute) softkey, and (RETURN). This starts the measurement. The state/software analyzer will begin storing states in memory and the timing analyzer will begin storing timing information. The state/software analyzer will enable the timing analyzer to search for its trigger (glitch) during the I/O driver routine, and will disable trigger search at all other times. When the timing analyzer finds its trigger, it will capture timing information and send the trigger transition to the state/software analyzer. The state/software analyzer will recognize this trigger and complete its trace of states. Now you can look at the states that led up to the error event and the states that were executed after the error occurred, as well as the hardware activity captured by the timing analyzer.

g. Rerun the test several times to see if the error occurs at the same point in software execution every time.

Chapter 8E
ASSERT FUNCTIONS

INTRODUCTION

This chapter discusses how you can use the state/software analyzer to assert the stimulus and halt lines to the preprocessor under selected signal conditions. You can also set up the state/software analyzer to assert BNC port 1 and BNC port 2 on the mainframe rear panel when these same signal conditions occur. These asserted lines can drive functions in a system under test and/or trigger associated measurement instruments. General information which applies to the asserted signals is discussed first. Then each of the signals is discussed in detail.

GENERAL INFORMATION

Each signal channel is activated and deactivated by use of the [assert] softkey in the trace specification. The [assert] softkey also allows you to make [polarity] selections for the signals from BNC ports 1 and 2. The polarity you select sets both BNC ports for positive-true or negative-true.

If you are operating a state/software analyzer in a system which has more than one analyzer installed, select only one of the analyzers to drive each BNC port on the rear of the mainframe. You cannot have two analyzers driving the same BNC port at the same time during a test. If this is attempted, the display will show an error message to warn you of an invalid setup.

THE STIMULUS LINE AND BNC PORT 1

The state/software analyzer can place a 50-nsec, TTL pulse on the stimulus line and/or BNC port 1 when selected signal conditions occur. The principle use of this pulse is to simulate an interrupt request to the system under test. You can assert the stimulus line to the preprocessor interface, or assert BNC port 1 on the mainframe rear panel, or assert both at the same time. The preprocessor will use the stimulus input to generate a 5-usec pulse or TTL level, as selected in the preprocessor specification. You can connect the stimulus pulse from the preprocessor interface or the 50-nsec pulse from BNC port 1 to the interrupt lines in your system under test.

You can select any of the following conditions to assert the pulse on the stimulus line and/or BNC port 1:

a. You can set up the trace specification to assert the pulse each time the state/software analyzer trigger specification is satisfied during a measurement execution. Then you can select trigger states that are within the areas where you want interrupts to occur.

b. You can set up the trace specification to deliver a pulse each time selected events occur in your sequence specification. Then you can set up the sequence with terms that occur at desired points in program flow.

c. You can set up the trace specification to deliver a pulse each time selected events occur in one of your window specifications. Then you can set up that window specification with the points in program flow where you want interrupts to occur.

When you are delivering the stimulus line pulse through a preprocessor interface, you can obtain an additional capability: you can have the system under test handshake the interrupt; that is, you can have the 50-nsec pulse set a latch in the preprocessor. The latch will maintain the true state of the interrupt request until the system under test returns an interrupt acknowledgement to the preprocessor. Then the preprocessor will reset the latch (interrupt request false). Refer to the state/software analyzer preprocessor specification for details of how to use this capability.

STIMULUS LINE COMMAND SYNTAX

a. [assert] [stimulus] [_on_] [triggers]. Each time your trigger specification is satisfied, the state/software analyzer will assert a pulse on the stimulus line to the preprocessor. You can select sequence terms or window terms instead of triggers, if desired.

b. [assert] [stimulus] [and] [bnc_1] [_on_] [triggers]. Each time your trigger specification is satisfied, the state/software analyzer will assert a pulse on the stimulus line to the preprocessor interface and assert the same pulse on BNC port 1 on the mainframe rear panel. You can select sequence terms or window terms instead of triggers, if desired.

c. [assert] [bnc_1] [_on_] [triggers]. Each time your trigger specification is satisfied, the state/software analyzer will assert a pulse on BNC port 1 on the mainframe rear panel. You can select sequence terms or window terms instead of triggers, if desired.

d. [assert] [stimulus] [and] [bnc_1] [off]. This command turns off the assert function to both outputs. You can turn off the assert function to either output, if desired. In this case, the other output will still be asserted according to your specification.

e. [assert] [polarity] [positive]. This selects generation of a positive pulse output on BNC port 1 when the specified conditions occur. You can enter negative instead of positive if you want the state/software analyzer to generate a negative pulse. This selection also applies to the output on BNC port 2, if asserted by this state/software analyzer.

THE HALT LINE AND BNC PORT 2

The halt line to the preprocessor interface can be asserted with run/halt TTL logic levels. From the preprocessor, this logic level can

be connected to the run/halt pin of the microprocessor in the system under test. The run/halt logic level can also be asserted to BNC port 2 on the mainframe rear panel.

The purpose of the halt line is to enable the state/software analyzer to halt operation in a system under test. You can halt system operation either when the state/software analyzer completes a measurement, or when it recognizes its trace point (first trigger occurrence) in program flow. The programmable halt can be used to page through program flow or to release the system until it reaches an area of interest in the state flow.

HALT LINE COMMAND SYNTAX

a. (assert) (halt_line) (_on_) (meas_comp). When the state/software analyzer completes its present measurement, it will assert the halt line in the preprocessor interface.

b. (assert) (bnc_2) (_on_) (meas_comp). When the state/software analyzer completes its present measurement, it will assert BNC port 2 on the mainframe rear panel.

c. (assert) (halt_line) (and) (bnc_2) (_on_) (meas_comp). When the state/software analyzer completes its present measurement, it will assert the halt line in the preprocessor interface and BNC port 2 on the mainframe rear panel.

d. (assert) (halt_line) (_on_) (trace_pt). When the state/software analyzer recognizes its trace point, it will assert the halt line in the preprocessor interface.

e. (assert) (bnc_2) (_on_) (trace_pt). When the state/software analyzer recognizes its trace point, it will assert BNC port 2 on the mainframe rear panel.

f. (assert) (halt_line) (and) (bnc_2) (_on_) (trace_pt). When the state/software analyzer recognizes its trace point, it will assert the halt line in the preprocessor interface and assert BNC port 2 on the mainframe rear panel.

g. (assert) (halt_line) (and) (bnc_2) (off). This command turns off the assert function to both outputs. You can turn off the assert function to either output, if desired. In this case, the other output will still be asserted according to your specification.

h. (assert) (polarity) (positive). This selects a rising edge halt output on BNC port 2 when the selected signal conditions occur. You can enter negative instead of positive if you want a falling edge halt output. This selection also applies to the output on BNC port 1, if asserted by this state/software analyzer.

Chapter 8F
MASTER ENABLE FUNCTION

INTRODUCTION

The master enable function provides a unique and powerful capability for masking undesired activity during measurements. In effect, master enable suspends time during undesired software executions. The measurement results in the trace memory and the overview memory appear to have been taken from a system that never executed the undesired activity.

USING MASTER ENABLE

The master enable function provides an additional, high level of control over the state/software analyzer. The state/software analyzer can be set up with specifications for a trace measurement, and other specifications for an overview measurement. Both the trace and overview measurements can be made independently from each other. Specifications for each can include different software windowing.

Master enable provides an additional level of windowing, above both measurement functions. When master is in the "enable" state, all of the elements of the state/software analyzer function according to their own specifications. When in the "disable" state, master suspends the trace measurement function, the overview measurement function, the sequence and window elements (except the sequence or window element assigned to master enable), and the time and/or state count function.

The default condition for master enable is (master) (enable) (always). If you activate master enable and operate it from either the sequence element or one of the window elements, it will enable the state/software analyzer only during periods when the associated sequence or window element is supplying its "enable" output.

When you use master enable to window a measurement, the trace memory will capture "next state" disable and enable, and the overview memory will capture "current state" disable and enable. There is a delay between detection of the disable or enable state by a sequence or window element and the time when the master function actually suspends or resumes analysis operations.

This delay allows the state/software analyzer to capture the state in trace memory that disables the master function. This delay also prevents the capture of the state that enables the master function.

There is a similar delay in the overview measurement operation. It occurs when the overview measurement classifies each state before storing the appropriate classification into overview memory. Because of the overview delay, the classification for the state that disables master will not be stored in overview memory, and the classification for the state that enables master will be stored.

Master enable can also be supplied to the intermodule bus during intermodule measurements. In these applications, operation is suspended in every receiving analysis module when master is disabled, and resumed when master is enabled. If you set up the state/software analyzer to receive master enable from the intermodule bus, then all functions, including all three sequence/window elements, will be suspended during the disable periods. Refer to chapter 8D of this manual for details.

The sequence or window element used by the master function cannot be used by any other activity in the state/software analyzer. This ensures that all other functions are disabled in the state/software analyzer during master disable periods.

EXAMPLE: MASKING INTERRUPTS DURING A MEASUREMENT

You can set up the master enable function to suspend operation in the state/software analyzer at the start of each interrupt processing routine and resume operation after completion of the interrupt process. You might set up window one to identify the interrupt process, and then master enable on window one:

(window) (one) (disable) (ADDRESS) (==) <address of entry to interrupt processor>

(enable) (ADDRESS) (==) <address of return to main program>

(master) (enable) (window) (one)

When you execute a measurement using the above master enable setup, the state/software analyzer will begin as though master enable does not exist. When window one detects the entry address of the interrupt processor, it will switch to disable. The master function will detect this and switch to disable, also. All measurement functions including time counts, state counts, the sequence, and window two will be suspended. Window one will continue to examine each incoming state until it finds the address of the return to main program. Then it will enable. Now master will switch to enable, allowing all of the analyzer measurement functions to resume their operations according to their individual specifications.

At the end of the measurement, the information captured in memory (including time measurements and/or counts of states) will have the interrupt activity masked out. (Because of the delay between detection of the entry to interrupt and master enable suspension of the trace measurement, the first state of the entry to interrupt will be captured in trace memory.) If a priority interrupt cycle occurs prior to entering the interrupt routine, detection of the interrupt can be used to completely window interrupt opcodes.

EXAMPLE: CAPTURING ONLY PASS TWO IN A THREE-PASS SOFTWARE EXECUTION

You may be analyzing a system that requires three passes through the software to complete a task. You may only want to analyze the activity that occurs during the second pass through the software. You might choose window two to identify the second pass, and then master enable on window two:

(window) (two) (enable) (ADDRESS) (___≡___) <beginning of pass two>

(disable) (ADDRESS) (___≡___) <end of pass two>

(master) (enable) (window) (two)

Window two will always be active during this measurement, but all of the other state/software analyzer measurement functions will be suspended (after the (execute) key is pressed). When window two finds the address which begins pass two, it will switch to the enable state. The master function will detect this change and release the state/software analyzer measurement functions to perform their own operations. Window two will continue to examine each state until it finds the address which ends pass two. Then it will disable again. This will be detected by the master function, and it will again suspend all measurement activity.

At the end of this measurement, the content of the trace and overview memories will appear to have been obtained from a system that executed a continuous series of "pass two's".

Chapter 9
THE PREPROCESSOR SPECIFICATION

INTRODUCTION

The preprocessor specification is where you can allocate the stimulus_line signal generated in the state/software analyzer. You can send this signal to your preprocessor interface, or to a BNC port on the rear panel of the mainframe, or to both destinations at the same time (as defined in the trace specification). The stimulus_line signal is discussed in the following paragraphs. This chapter also describes the softkey selections available for using this signal resource.

You can only get this specification on screen if your state/software analyzer is equipped with a preprocessor for which a file of the name "state<ID>:HP:system" exists (where <ID> is the ID code of the preprocessor). The file "state80:HP:system" must also be present. Refer to the preprocessor manual for a description of where to find the stimulus line and how to connect it in the preprocessor.

THE STIMULUS LINE

The stimulus line carries a pulse with a duration of about 50 nanoseconds. This pulse is generated under a variety of state flow conditions which you can select in the trace specification. The preprocessor outputs a pulse of about 5 microseconds each time it receives the stimulus pulse.

STIMULUS PULSE GENERATION

In the preprocessor specification, you can allow a [repeat] of the stimulus_line pulse each time the trace specification is satisfied, or only a [single] generation of the stimulus_line pulse the first time the trace specification is met during a trace.

STIMULUS PULSE MODES

You can select either the [pulsed] or [hand_shk] mode to be carried out in the external preprocessor. If you choose [pulsed], the stimulus_line pulse will pass through the preprocessor the same as it does through the rear-panel BNC. If you choose [hand_shk], the stimulus_line pulse will set a latch in the preprocessor interface, and the interface will produce a steady-state output. The acknowledge line from the target system must be connected to reset the latch in the preprocessor (handshaking the stimulus level).

STIMULUS_LINE APPLICATIONS

The principle use of the stimulus_line is to assert an interrupt at a specified point in program flow. You can either connect a line from the preprocessor to the interrupt input of your target microprocessor, or from BNC port 1 on the analyzer mainframe. The (hand_shk) mode lets you include the acknowledge line from the target system in your interrupt processing, if the preprocessor is used.

You can use the stimulus_line pulses from BNC port 1 to control measurements in associated equipment. By setting up your trace specification to generate a pulse every time a particular event occurs, you can watch corresponding electrical waveforms on an oscilloscope, or count events on a counter.

PREPROCESSOR SPECIFICATION SOFTKEYS

The softkeys in this display provide the functions described below when you are using a dedicated preprocessor and you have asserted the stimulus line in the trace specification.

(stimulus) (disabled) - The stimulus line is inactive when this specification is selected.

(stimulus) (repeat) (hand_shk) - The stimulus line will send a pulse to the preprocessor every time your trace specification requirements are met. This command causes the preprocessor to retain a set state on its stimulus output until it is reset by an acknowledgement from the target system.

(stimulus) (repeat) (pulsed) - The stimulus line will send a pulse to the preprocessor every time your trace specification requirements are met.

(stimulus) (single) (hand_shk) - The stimulus line will send a pulse to the preprocessor the first time your trace specification requirements are met. The preprocessor will retain a set state after it receives the pulse until it is reset by an acknowledgement from the target system. Then the stimulus_line will be disabled for the remainder of the trace.

(stimulus) (single) (pulsed) - The stimulus line will send a pulse to the preprocessor the first time your trace specification requirements are met. Then the stimulus_line will be disabled for the remainder of the trace.

The remainder of the softkeys available in this display are the utility keys discussed in Chapter 5 of this manual.

Chapter 10
TRACE LIST

INTRODUCTION

The state/software analyzer uses three forms of trace list to present information in its trace list display. The first form of display is a list of states captured in the trace memory according to the specified measurement. The second and third forms of trace list are only available in state/software analyzers equipped with one or two memory expanders. The second form is a list of statements from the source files that created the trace data. The third form is a combination of forms one and two: source statements followed by the trace data that was emitted by those statements. The source statements are normally shown in inverse video, and the trace data is shown in normal video.

The trace list also shows analyzer-induced comments. The analyzer will comment each point where storage was disabled and enabled during a measurement (enable on window, sequence, inter-module bus, etc). Analyzers equipped with memory expanders will comment before each source line or block of lines to show the source file name, and high-level line number(s). In the display of source-only statements, the analyzer will additionally show the trace memory address where the first byte/word emitted by that statement is located, and the absolute time between the trigger and first byte/word emitted by that statement.

The trace memory is 256-locations deep. It stores each state that meets the store specification along with counts of time or states and the status of the sequence and window elements. Softkeys presented on the trace list let you format the information from the trace memory in a variety of ways.

The state/software analyzer automatically formats the trace list during initialization. The arrangement of information in the trace list depends upon which interface is connected. If you are using the general-purpose preprocessor with a dedicated interface, the trace list will be formatted with columns of information labeled to correspond to the type of processor system under test. If you are using general-purpose probes, the display will show labels for each of the probe pods. You can use the softkeys of the trace list to reformat the display, if desired.

During the time the state/software analyzer is formatting a trace list, it can access any of the maps you created and display symbols defined on them. If the analyzer is equipped with a memory expander, it can also look up the contents of the link_symbols table for the absolute file and display symbols and segment names defined there. If your analyzer is not equipped with a memory expander, you can still obtain displays which show the symbol and segment names. Simply upload them into maps as described in Chapter 7. If you used descriptive names for your symbols, those names will help you in debugging the trace.

This chapter describes the trace list display during a trace measurement and after the trace measurement has been completed or halted. It also describes the softkeys available in the trace list, and shows you how to use them to arrange a display of the desired information. This chapter shows the types of information you can get on a trace list, and how to interpret the content of each column on your display. The end of this chapter shows several examples of how to use trace list displays.

TRACE LIST DISPLAY INTERPRETATION

Figure 10-1 shows a typical trace list display. It is included in this chapter to show you how to read the information shown on a trace list.

The top line of the display shows that this trace list was obtained by the state/software analyzer whose control board is installed in mainframe slot 3. It is a 60-channel state/software analyzer, and is collecting data through an 8085 interface.

The first column on the display shows the area of memory that is being displayed (the trigger state plus the next thirteen states that were stored in memory).

The default trace list display is shown in Figure 10-1. It is obtained by pressing (execute) and (RETURN). This is the display obtained with (source) (off).

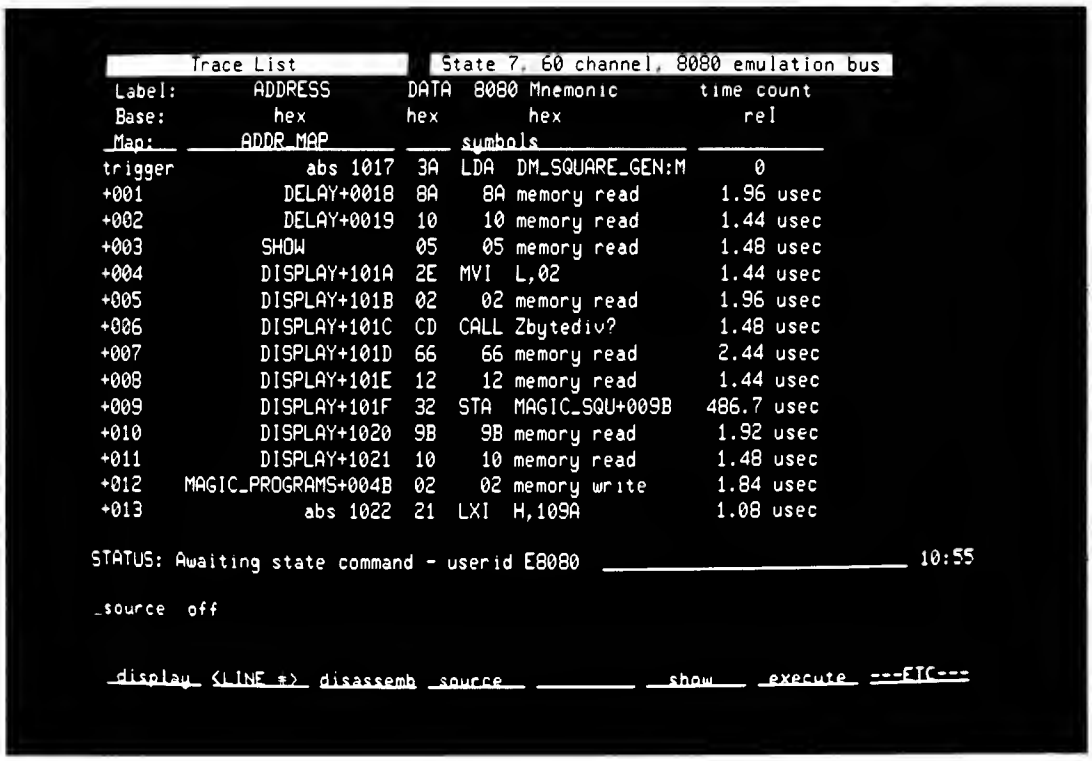


Figure 10-1. Typical List of Trace Data (source) (off)

The next column lists activity collected from the probe set which was assigned the ADDRESS label. The values shown are relative to the symbol map called ADDR_MAP (for those values with symbols assigned). For values which have no symbols assigned, absolute numbers are shown. All numerical values in this column are in the hexadecimal number base.

The mnemonic column shows that its information was derived by using the inverse-assembler file for 8085. All numerical values within this column are in the hexadecimal number base.

The time count column shows the relative time between each of the states captured in memory.

The DATA column shows information captured from the probe set which was assigned the DATA label. It shows no map name; therefore all of its information is shown in absolute values.

The STATUS line shows that the state/software analyzer is not performing any activity. It is ready to accept new commands from the keyboard.

Figure 10-2 is only available to a state/software analyzer equipped with a memory expander. You can obtain this display by pressing the following keys:

(source) (only) (RETURN)

This display contains two types of information: source file lines (shown in inverse video), and analyzer-induced comments (shown in normal video, beginning and ending with strings of ASCII pad characters).

The analyzer-induced comments contain the line number of the first byte or word emitted by the next source statement. The comment offers information about the source statement, such as the name of the source module where it resides. Following that is another line number. This is the line number of the source statement in the source file. Finally, the time count column shows the elapsed time between the trigger state and the first state emitted by the source statement.

Figure 10-3 shows a mixed display of high-level source statements and trace data. As with Figure 10-2, a memory expander is required to obtain this display. Press the following keys to obtain the display shown in Figure 10-3.

(source) (on) (RETURN)

High-level source statements (shown in inverse video) are placed on screen before the code emitted by them (shown in normal video). Up to 31 lines of information (source statements and analyzer-induced comments) may precede a state of trace data on the display.

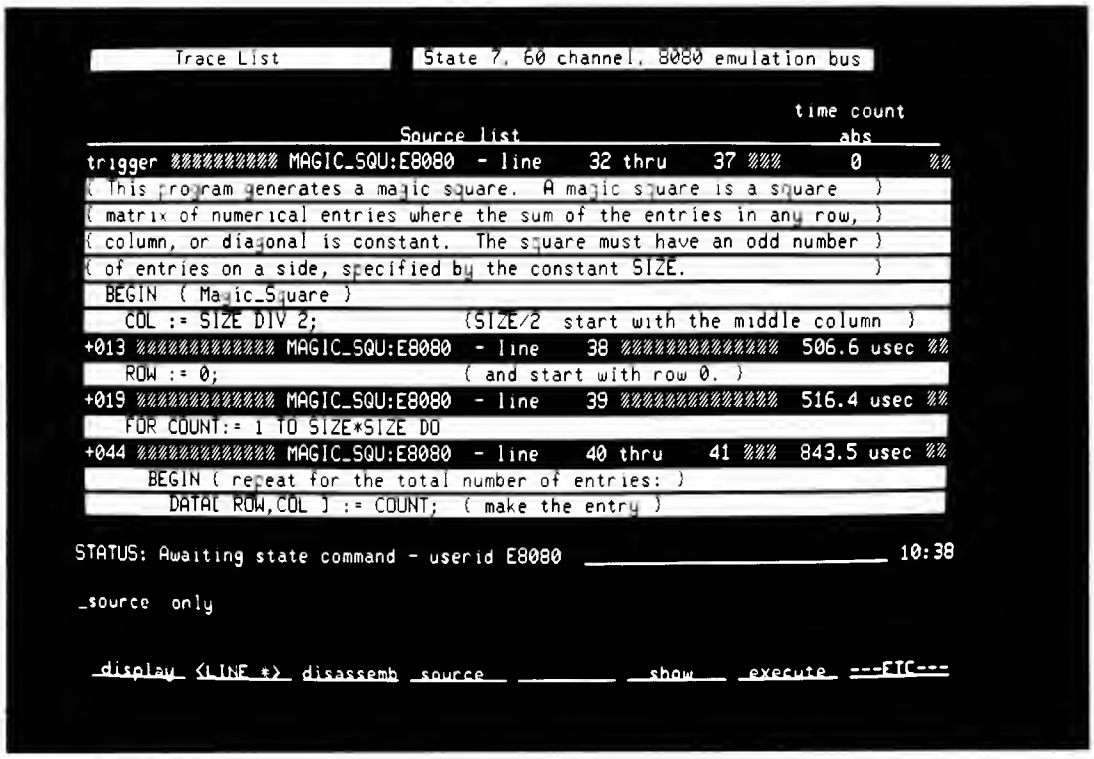


Figure 10-2. Typical Display Of High-Level Source Statements (source) (only)

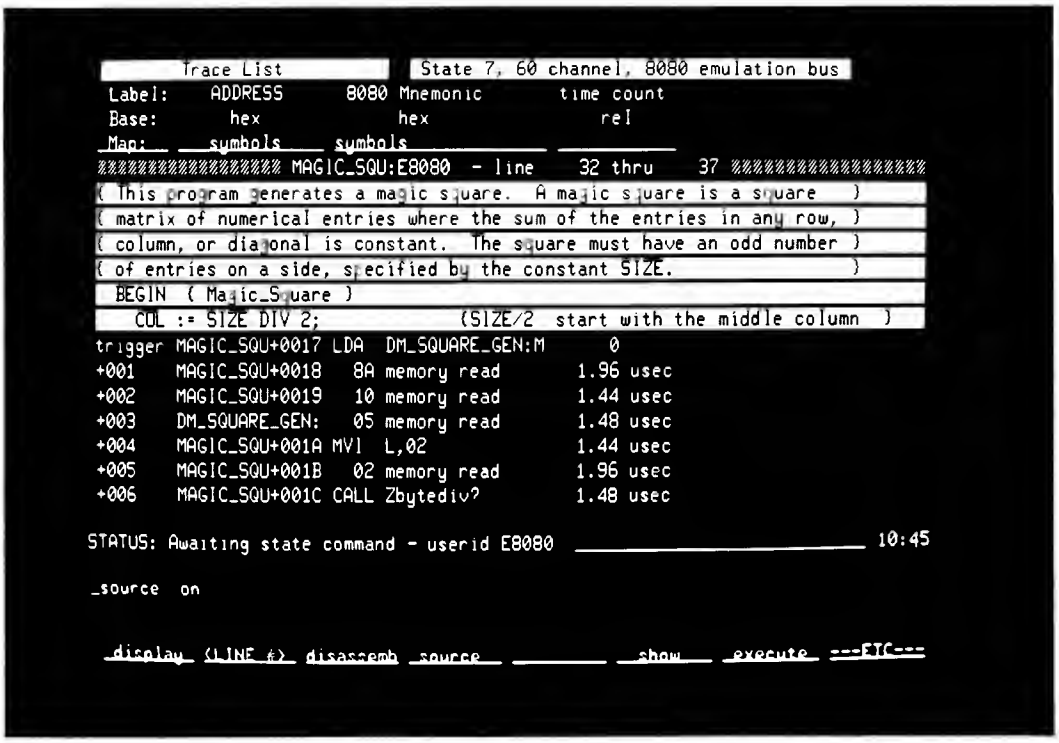


Figure 10-3. Typical Mixed Display of High-Level Source Statements and Associated Trace Data (source) (on)

TRACE LIST DURING THE MEASUREMENT

While the run is in process, the status line will show the status of the analyzer. It may display any of the following messages as information for the operator:

Waiting for trigger - This message indicates the trace has started. Any states that meet the "store" specification are being stored in memory. The trigger state has not yet been found.

Trace in process - This message indicates the trace has started and trigger has been found. Not enough states have been stored yet to fill the trace memory.

Overview in process - This message indicates the trace has been completed, but the overview measurement is still incomplete. Refer to the chapter on overview displays for detailed information.

Slow Clock (inverse video) - This message indicates that the rate of the incoming clock (if any) is below the specified clock rate of the state/software analyzer.

The state/software analyzer will interactively read the trace memory and update the display while a trace is in process. If the clock rate is too fast, the trace memory content may not be displayed during the trace. Display of interactive data is guaranteed with clock rates up to 4.75 MHz.

Multiple module executions have a separate set of messages, as follows:

Single (or Multiple) module execution in process (or completed, or halted) - This message describes the status of the measurement in which the state/software analyzer is taking part.

Module Not Involved - This message indicates that the multiple module execution is in process, but the state/software analyzer is not included in the measurement.

TRACE LIST AFTER THE MEASUREMENT ENDS

When a normal trace measurement ends, the status line will show "Trace complete". The display will contain a list of the states obtained during the measurement. The default list will show the portion of memory that includes the trigger state. If you have positioned the display to show any other part of the trace memory, that part of memory will be displayed with each new trace execution until you reposition the display again.

You may execute trace measurements which do not completely fill the trace memory. During a measurement which is acquiring information at a slow rate, you may press the (halt) key and (RETURN). In this case, your measurement will end. You can obtain a display of the entire memory content, but blank lines will be displayed where no new states were captured during the trace.

Another common cause for not completely filling the trace memory during a measurement is the trigger position specification. When the state/software analyzer executes a trace measurement, it examines each incoming state to see if it meets its trigger specification. At the same time, it stores each state in memory if the state meets the store specification. When it finds the trigger state, it stores enough additional states to fill the memory following the specified position of your trigger.

For example, if you specified trigger position as "center of trace" and the first state found by the analyzer met the trigger specification, that state would be stored as the trigger. Then the next 127 states that met the store specification would also be stored, and the trace measurement would end. This would leave lines -128 through -1 blank.

The notation "history" may also appear in your trace list. It is shown on the display when you halt your trace before the trigger state has been found. The "history" notation is always the last entry in the trace memory. The remainder of the memory will contain activity that was stored before you halted the trace.

TRACE LIST SOFTKEYS

There are four softkeys on this display that have special applications in the trace list: (display), (disassemb), (source), and (copy). Use of these four keys will give you control of the power of the trace list display. The state/software analyzer will answer these key entries both during a trace and after the trace has been completed. Each of these softkeys is discussed in detail in the following paragraphs:

DISPLAY

The information in the trace list is presented according to the default display arrangement. If you are using a general-purpose preprocessor with dedicated interface, the presentation will show information related to the system you are testing. If you are using general-purpose probes, the display will list the probe pods and show the hexadecimal values collected from each one.

The [display] softkey is used to select a new arrangement of the columns of information on the display, and to specify the format of the information that is presented in each column. Each of the selections that can be made with the [display] softkey are discussed in the following paragraphs:

1. [display] [modify] - This entry calls the present display arrangement to the command line. Use this entry when you want to rearrange the order and/or content of the information presented in the trace list.

2. [display] [mnemonic] - This entry places a mnemonic list on screen. The list is obtained from the inverse-assembler which uses a disassembler relocatable file such as I8085;HP:reloc to interpret the activity captured from the system under test. If your system does not have the inverse-assembler relocatable file, the STATUS line will show "No disassembler" when you try to obtain this function. You can include the display of [symbols] and/or [segments] from the absolute file and MAP symbols together in columns of mnemonic information. Refer to <LABELS> paragraph 8 for details.

3. [display] [line_numb] - This entry calls a column showing the numbers of the associated high-level statements in the source file. The line numbers from the source file will be shown beside the first byte or word emitted by each statement, and beside each additional opcode emitted by the statement. The number beside the first state will have an asterisk (45* identifies the first state emitted by line 45 in the source file; 45 without the asterisk identifies another opcode which was also emitted by line 45 in the source file).

4. [display] [count_abs] - This entry calls a column of absolute counts to the display. It shows the count of states or measure of time between the trigger state and each state stored in trace memory.

5. [display] [count_rel] - This entry calls a column of relative counts to the display. It shows a count of states or measure of time between each of the states stored in trace memory.

6. (display) (sequence) - This entry calls three columns to the display. These three columns show the status of the sequence and two window elements when each of the states was captured.

Example:

	seq	win 1	win 2
"find 1"			
"find 2"	1d	d	off
"find 3"	2d	d	off
enable	2d	e	off
"find 4"	2d	e	off
"find 5"	3d	e	off
disable	4e	e	off
	4e	d	off
	5e	d	off
	1d	d	off
	1d	e	off

In this example, window 2 was not used. The specification for window 1 was loaded with a state on which to enable and a state on which to disable. The example shows that window 1 began in the disable state. It did not find its "enable" state the first time, but it did find it the second time. The second "d" indicates that the window 1 element was still providing its disable output when the corresponding state was captured. The third entry is an "e". This indicates that the second state must have satisfied the enable specification for window 1 because when the third state was captured, window 1 had switched to the "enable" output. The "d" or "e" in the column indicates the output condition of the element when the associated state was captured in memory. It does not indicate whether the captured state satisfied any part of the specification for window 1.

The column under seq (the sequence element) shows both a letter and a number. The letter "d" or "e" indicates the present form of the output from the sequence element (exactly the same as in the columns for the window elements). The number beside each letter shows which one of the sequence term numbers the state/software analyzer was looking for when the associated state was captured. The number in the next line will tell you whether the search for term number was successful or not. If the sequence element is looking for a new term number in the next line, it is because the previous search was successful.

7. (display) (space) <NO.> - This entry is used to change the spacing between adjacent columns to aid readability. The default of (space) is 0 between columns.

8. (display) <LABELS> - You can call for display of the activity captured from any labeled set of probe leads. The analyzer allows you to specify display of the information captured from a label in absolute values in any of the four number bases ((display) (ADDRESS) (in_hex) (absolute)), or in the ASCII character set. You can also have the information displayed using symbols and offset values from any of the symbol maps. Your offset values can be in any of the four number bases or ASCII ((display) (ADDRESS) (in_hex) (relative) (ADDR_MAP)). The default display ((display) (ADDRESS)) will present your label information using symbols relative to the default symbol map for that label, and offset values in the hexadecimal number base. If you have identified your absolute file ((absolute) <FILE NAME>), the analyzer will also show symbols and segments from the link_symbols file.

If the analyzer is equipped with a memory expander, it can also show symbols and segment names (code module names) recorded in the link_symbols table in the same column with the map symbols. The analyzer executes the routine shown in Figure 10-4 to determine which symbol (from a map or the absolute file) to present on screen to represent each value in a trace list.

7. (then) - This softkey is used to add columns of information to the trace list display. When you have completed the specification for a column of information, the (then) softkey allows you to enter a specification for the next column to be adjacent to the first column.

(display) (ADDRESS) (then) (DATA) (then) (count_rel)

The above command line will place three columns on the trace list: the first will have information from the probe lines labeled ADDRESS, the second from the probe lines labeled DATA, and the third column will have the measures of time (or counts of states) between each of the states shown in the list. The values under ADDRESS and DATA will be hexadecimal numbers (the default number base). They may be absolute or symbols relative to a symbol map (if you assigned a default map to support one or both of the labels).

(not_block) - This softkey deletes the lines that the analyzer uses to identify periods where storage was disabled during a trace. When you specify storage other than (store) (enable) (always), you may have contiguous states in memory that were not executed sequentially in the system under test because storage was disabled during periods between states. The analyzer remembers periods when storage was disabled and identifies those periods by placing the following line on screen between blocks of states:

[] [] [] [] [] [] [] store disable/enable [] [] [] [] []

The (not_block) softkey is useful in cases where the block identifier lines occupy a large portion of screen space (such as when you specified store-enable on window one and window one disabled and enabled so often that every other line on screen is a block identifier).

[rolled] - This softkey is used to roll the content of a column up or down by a selected offset. This can be used to compare successive executions on a label by making two columns of the same label and rolling one column the number of spaces required for one execution.

[width] - This softkey allows you to specify a truncation of a column to make room for additional columns on the screen.

[absolute] - This softkey allows you to specify that each value in a column be shown in terms of absolute values in any base of your choice (hexadecimal, if unspecified).

[relative] - This softkey allows you to specify that each value in a column be shown relative to symbols defined on a map or symbols and/or segments defined in the absolute file, or both. Refer to paragraph 8 for details.

DISASSEMBLE

The [disassemb] softkey allows you to control the inverse-assembler which provides mnemonic information to the trace list display. If you are using a general-purpose preprocessor with a dedicated interface, your software will already have the correct configuration file and inverse-assembler file for the microprocessor system you are analyzing. The [disassemb] selections will all be made automatically during initialization.

If you are using general-purpose probes, you will have to make the [disassemb] entries so that the proper probe lines will be monitored for address, data, and status information and the correct inverse-assembler file will be used to obtain mnemonic displays. The [disassemb] entries you will need are discussed in the following paragraphs:

1. [disassemb] [using] <FILE> - This causes the disassembler to use the inverse-assembler file of your choice to interpret information it receives at its input. If you enter [disassemb] [using] I8085 [RETURN], the inverse assembler file labeled I8085:HP:reloc is used to obtain mnemonic displays of information captured by the probes.

2. [disassemb] [from_line] <LINE> - This instruction can be used to synchronize your inverse-assembler to begin at a particular line of the trace memory. The display will be positioned to place your line number as the second line of information on the screen. The inverse assembler uses the line number you enter for status synchronization (opcode) when insufficient status is provided. Insufficient status may occur when you are probing with general-purpose probes.

3. [disassemb] [address_is] <LABEL> - This instruction defines which labeled set of probes will supply information to the first disassembler input field (called "INPUT_ADDRESS"). Normally the information in this field is address information from the system under test.

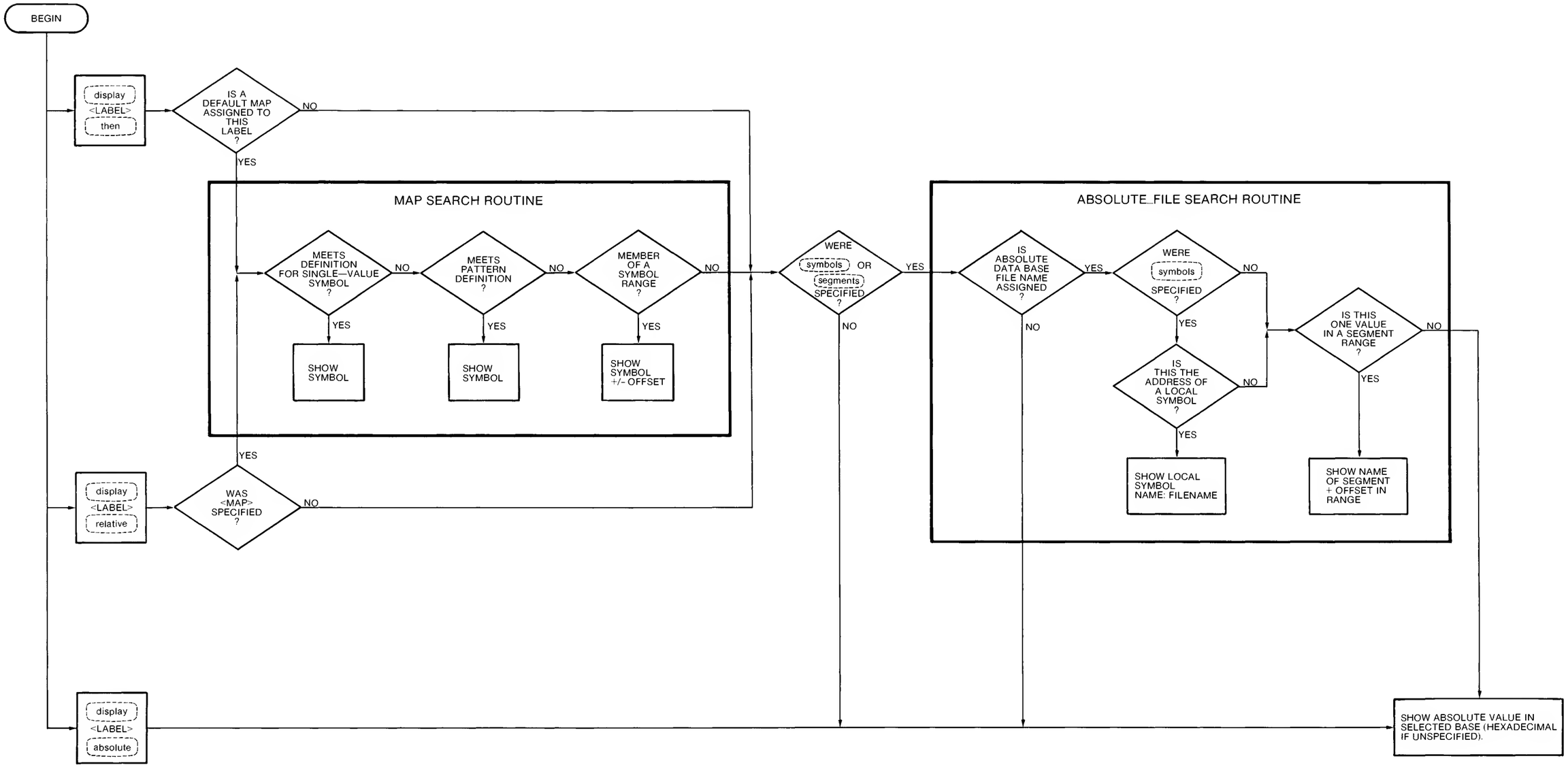


Figure 10-4.
Trace List Symbol Search

4. (disassemb) (data_is) <LABEL> - This instruction defines which labeled set of probes will supply information to the second disassembler input field (called "INPUT_DATA"). Normally the information in this field is data from the system under test.

5. (disassemb) (status_is) <LABEL> - This instruction defines which labeled set of probes will supply information to the third disassembler input field (called "INPUT_STATUS"). Normally the information in this field is status from the system under test.

The disassembler has the 'address_is, data_is, status_is' input fields. Each of these input fields accepts activity from a labeled set of input lines. Each labeled set can be up to 32 bits wide. With maximum inputs in each of the three fields, the disassembler will receive states that are 96 bits wide.

In the normal case, the disassembler will interpret the activity within the first field as address information, the activity in the second field as data information, and the activity in the third field as status information. By assigning a label to each of these three fields, you determine which probe lines will be read by the disassembler to obtain activity in each field.

Refer to the operator's manual for the preprocessor interface that you are using. It describes the required formatting to obtain inverse-assembly.

SOURCE

The (source) softkey offers the following capabilities in state/software analyzers equipped with memory expanders. It allows you to select between the three forms of trace list display, as follows:

1. (source) (off) - selects trace data displays (such as Figure 10-1).

2. (source) (only) - selects lines from the high-level source file that emitted the trace data in memory, along with analyzer-induced comments (such as Figure 10-2).

3. (source) (on) - selects displays combining 1 and 2, above (such as Figure 10-3).

4. (source) (on/only) (non_inver) - This changes the display of the high-level source statements to normal video with ASCII pad characters at each end of the display width.

COPY

The (copy) softkey has the (copy) (tracelist) capability; it is in no other display. This allows you to obtain a copy of the information in trace memory beyond that shown on the display. You can copy (all) of the information in the trace memory, or just that portion of the memory which starts on the CRT and continues (thru) the (start) or (end) of the memory, or (thru) any line number of your choice. The state/software analyzer will not copy any blank lines in memory. The columns of information in your copy will be arranged in the order and with the content you specified with the (display) softkey. Your CRT may not have room enough to show all of the columns of information that you want in your copy. If your printer is wide enough, you will have all of the columns present on your hard copy even though they are not all on the CRT.

DISPLAY POSITIONING

The following keyboard keys control the position of the display window in the trace memory: (ROLL UP), (ROLL DOWN), (SHIFT) (⟶), (SHIFT) (⟵), (NEXT PAGE), and (PREV PAGE). These keys are discussed below:

(ROLL UP) — This key scrolls through the trace memory, line-by-line. It increments the line numbers.

(ROLL DOWN) — This key scrolls through the trace memory, line-by-line. It decrements the line numbers.

(SHIFT) (⟵) — These two keys together scroll the display one column at a time to the left to allow you to format and observe more than one screen width of data.

(SHIFT) (⟶) — These two keys together scroll the display one column at a time to the right and allow you to format and observe more than one screen width of data.

(NEXT PAGE) — This key steps the display window through the trace memory. Each new page contains the next higher set of line numbers from the trace memory.

(PREV PAGE) — This key steps the display window through the trace memory. Each new page contains the next lower set of line numbers from the trace memory.

You can also position the display to an area of trace memory by typing a number (line number) on the command line. When you press (RETURN), the display will place your line in the center of the screen, along with adjacent lines of information, if available.

NOTE

The trigger/history is always on line 0.

When (NEXT PAGE) is pressed, the display builds from the top down. When (PREV PAGE) is pressed, the display builds from the bottom up. When a line number is specified on the command line, the display builds both up and down from the center. The analyzer repositions its display this way because it begins building the display before it knows how many lines of trace data and high-level source statements will be presented on screen. The display is built one line at a time beginning at the designated reverence point (top line, bottom line, or center line) until the screen is filled.

When you are displaying high-level source code and you type in a line number, you might not get the exact number on screen. The analyzer will see if your line number is the first state emitted by a source statement. If it is, the line will be placed in the center of the screen. If not, the analyzer will search ahead in memory until it does find the first state beginning a new source file statement, and that line will be placed at the center of the screen. You may call for line 100 and get line 109 instead because line 100 did not begin a source statement, and line 109 was the first line after 100 that did begin a source statement.

EXAMPLES

The following examples demonstrate ways to use the softkeys to obtain information on a trace list:

DISPLAYING A LABEL IN SEVERAL BASES

You can get different decodings of the same labeled set all together on a display. Assuming you have assigned the label ADDRESS to a set of input channels, you can press the following keys:

(display) (===ETC===) (ADDRESS) (in_hex) (then) (===ETC===)

(ADDRESS) (in_oct) (then) (===ETC===) (_DATA_) (in_asc)

When you press (RETURN), the display will show three columns on screen. Each will show the same part of memory, two with information from the ADDRESS label, and one from the DATA label. The two ADDRESS columns will be in different number bases. The first ADDRESS column will use the hexadecimal number base, and the second will use the octal number base. The DATA column will show information using the ASCII character set. The analyzer calculates ASCII characters by dividing the memory content of the label into 8-bit frames and then reading the first seven bits for the equivalent character, and discarding the eighth bit (parity). If the label you specified is not equally divisable by 8, the last ASCII character will be determined by accepting the bits available and adding trailing zeros to obtain the required eight bits.

DISPLAYING A LABEL WITH MULTIPLE SYMBOL MAPS

Perhaps your address space is divided between two or more distinct functions, such as system operations, and application operations. You might want to use separate symbol maps to describe activity in each of these operations. You can call up a display of your address activity and use one column with symbols from the map of system activity, and the other column with symbols from the map of application activity. This can be obtained by a command line such as:

(display) (===ETC===) (ADDRESS) (relative) (SYS_MAP) (then)

(===ETC===) (ADDRESS) (relative) (APPL_MAP)

When you press (RETURN), the display will show two columns of information taken from the probes labeled ADDRESS. The values in the first column will be represented by symbols from the system map. The values in the second column will be represented by symbols from the measurement map. (This example assumes these symbol maps were defined in the map specification.)

DISPLAYING OVERLAPPING LABELS

For your measurement, you might have grouped the status lines together under the label STATUS. You might also have assigned a separate label to one of the lines in the status bus, such as RW for the read/write line. You can format a display which includes both the RW label and the overlapping STATUS label, as follows:

(display) (===ETC===) (STATUS) (relative) (STAT_MAP) (then)

(===ETC===) (_RW_) (absolute)

When you press (RETURN), the display will show two columns of information. The first column will represent the values obtained from the status bus using symbols from the STAT_MAP. The second column will show the condition of the read/write line in absolute numbers.

You can set up a symbol map to support the 1 and 0 values of the read/write line. Then you can modify the above command line to use these symbols under the RW label:

(===ETC===) (_RW_) (relative) (RW_MAP)

DISPLAYING A LABEL RELATIVE TO SYMBOLS AND SEGMENTS

Within your absolute file, you might have single-value symbols within code modules. You might like to see these symbols and identify their locations within the modules. The following command line will show you this information:

(display) (===ETC===) (ADDRESS) (relative) (symbols) (then)

(===ETC===) (ADDRESS) (relative) (segments)

When you press (RETURN), the display will show two columns on screen. Each column will show information under the ADDRESS label taken from the same area of memory. Both the 'symbols' column and the 'segments' column will list the segment names with corresponding offset values to show locations of transactions within the modules. The 'symbols' column will also show the names of any single-value symbols residing within the segments.

ADDRESS hex symbols	ADDRESS hex segments
Apply+0002	Apply+0002
Apply+0003	Apply+0003
Apply_production	Apply+0127

Chapter 11
OVERVIEW DISPLAYS

INTRODUCTION

Overview displays answer a need for information that is difficult to get from a trace list. They show patterns of program execution and usage of program time or activity instead of the details of state execution. Overview displays are obtained by pressing the (show) and (overview) softkeys, and pressing (RETURN).

The state/software analyzer can formulate overview displays from two memory sources: the trace memory, and the overview-event memory. From the trace memory, the state/software analyzer can compose graphic displays of the activity captured from any of the labeled sets of probe leads. From the overview event memory, the state/software analyzer can formulate histograms, lists, and graphs. The overview event memory resides on the 20-channel acquisition board. This board must be installed as pod 1 in order to obtain overview-event memory displays.

This chapter describes how to obtain and use the overview displays in the state/software analyzer. Overview graphs of information from the trace memory are discussed first because these overview displays are available in all installation configurations of the state/software analyzer.

OVERVIEW OF INFORMATION STORED IN TRACE MEMORY

From the states stored in the trace memory, the state/software analyzer can compose a graphic display of the activity captured in any labeled set of probe leads. These graphic displays show patterns of program activity. To obtain a graph of the activity captured from a labeled probe set, press the following softkeys:

(display) (graph_of) (ADDRESS) (RETURN)
↑
name of label you want to graph

HOW TO READ THE GRAPH OF A LABEL

With the display on screen, you will see a pattern of asterisks across the CRT. These represent values which were measured on the set of labeled probe lines. The default condition of the graph establishes a y-axis scale that accommodates all possible values from the labeled set of input bits. You might want to make the y-axis scale more narrow to expand details of the measurement. Press the following keys:

(scale) (decrement) (y_axis) (upper)

Now every time you press the (RETURN) key, the upper limit of the y-axis scale will be decreased.

(scale) (increment) (y_axis) (lower)

Every time you press the (RETURN) key, the lower limit of the y-axis scale will be increased. You can repeat these last two operations until you have the activity from the labeled set of probe lines spread out over the entire range of vertical display.

The default condition of the horizontal scale (x axis) shows 64 consecutive memory locations across the screen. You can change the horizontal scale of the display to include more memory locations or to reduce the number of memory locations on the graph, if desired. Use the (scale) softkey, as follows:

(scale) ((increment) or (decrement)) (x_axis) (upper)

AND/OR

(scale) ((increment) or (decrement)) (x_axis) (lower)

There is another way to obtain specific scaling for the x-axis and y-axis parameters. If you know the exact dimensions you want for your x-axis (line numbers) and y-axis (values) scaling, you can make these entries as follows:

(scale) (x_axis) 168 (_to_) 255 (RETURN)

AND

(scale) (y_axis) 1A8H (_to_) 1C1H (RETURN)

The above commands will establish a graph that shows the trace memory content of lines 168 through 255 across the X axis and distribute the values from those memory locations across the Y axis from 1A8H to 1C1H.

There is interaction between the trace list display and the overview graph of a label. The upper and lower limits of the x-axis are defined by line numbers. The number that defines the lower limit on your x-axis display is the number of the line that will be at the top of your display if you switch to a trace list display. The ROLL keys on the keyboard will scroll your graph through the content of the trace memory just as they scroll through the states in a trace list display.

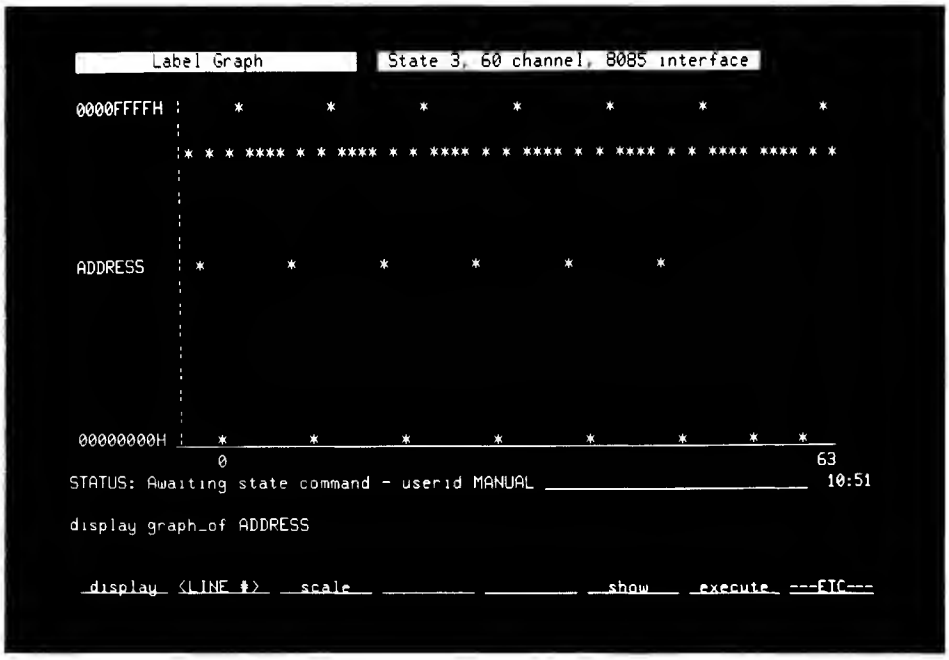


Figure 11-1. Typical Graphic Display Of Label Activity

SOFTKEYS

The (display) and (scale) softkeys have unique applications in overview displays. You can obtain the control features offered by these two softkeys when a measurement is in process as well as after the measurement has been completed or halted.

DISPLAY

Use the (display) softkey to select one of the four types of overview displays. Three of these displays are composed of information from the overview-event memory, one from information in the trace memory. The following sample command lines show how to use the (display) softkey.

(display) (graph_of) <LABEL> - This display shows a graph of the state activity found in the labeled set of input probes that you select. This information is obtained by the state/software analyzer reading the trace memory. You can obtain a graph of activity of any of your labels. With this display, you can see patterns in the state flow. Refer to the paragraph on how to read the graph of a label.

(display) (overview) (histogram) - This display shows all of the events from the event memory on a bar graph. The length of each bar on the histogram is proportional to the amount of activity that was found to meet the specification of the associated event during the most recent measurement. The display underlines the event which had the most activity.

NOTE

To display (overview), the overview function must be turned on in the trace specification.

The histogram display includes two numerical columns between the event names and bar graph: Cnt. and %. These two columns list the total number of occurrences that met the specifications for each of the events, and what percent of the measurement was found to satisfy each of the event specifications.

There are two softkeys which can be added to this command to expand the scale of the display: (max_peak) and (flt_peak). If you select either of these softkeys, the longest bar on the histogram will be expanded to a full-scale display. The other bars will also be expanded in proportion to the first. The horizontal scale will be adjusted in proportion to the expanded histogram. Refer to the paragraph describing the overview-event histogram.

(display) (overview) (list) - This display shows sixteen lines of the overview-event memory. You can scroll the 16-line window through any portion of the memory. The line number column shows the locations of each of the event codes. The Event column lists event names to show the order that each of the events was captured in memory. If the entire overview-event memory was not filled during the run, the Event column will be blank beside the unfilled line numbers. Refer to the paragraph describing the overview-event list.

(display) (overview) (graph) - This display shows all of the events from the event memory along the vertical axis, and represents activity found in these events by placing asterisks along the X axis. You can use this display to see patterns of program activity as execution transitions from one event to another. Refer to the paragraph describing the overview-event graph.

SCALE

This softkey is available in the two overview graph displays. With this softkey, you can scale your graphs to include any part (or all) of the memory contents being graphed. The (scale) softkey allows you to individually (increment) and (decrement) the (upper) and (lower) limits of the (x_axis) and (y_axis). scales. You can also enter direct values for the x-axis and y-axis limits. These values can be stated in absolute numbers in one of the four number bases; they can be mathematical expressions; they can be symbol values from any of the symbol maps; or they can be combinations of these.

NOTE

If the overview graph is scaled to include a large portion (or all) of the 4K overview memory, the state/software analyzer will have a long delay before showing the display.

(scale) (decrement) (y_axis) (upper) - This command sets up the state/software analyzer to decrement the upper limit of the vertical scale each time that the (RETURN) key is pressed. If you hold the (RETURN) key, the y-axis upper limit will continuously decrement until you release the key.

(scale) (increment) (x_axis) (lower) - This command sets up the state/software analyzer to increment the value of the lower limit of the horizontal scale. You can increment the value each time you press the (RETURN) key, or you can hold the (RETURN) key and the x-axis will continuously increment its value until you release the key.

(scale) (x-axis) 144 (_to_) 255 - This command sets up the state/software analyzer to use the specified upper and lower limits for the x-axis scale. This entry will begin the display with the information stored on line number 144 and end the display with the information from line number 255 in the trace memory. If your entry exceeds the limits of valid memory size (valid scaling limits), the state/software analyzer will default to the maximum and/or minimum values, as applicable. Limits can be stated in negative numbers when the trigger position is not start of trace.

(scale) (y-axis) 1A8H (_to_) 1C1H - This command sets up the state/software analyzer to use the specified upper and lower limits for the vertical scale. This entry will cause equal distribution of values from 1A8H to 1C1H across the vertical area of the CRT. If your entry exceeds the limits of scaling available on the vertical axis, the state/software analyzer will default to the maximum and/or minimum values, as applicable.

NOTE

Y-axis scaling is only adjustable when graphing a label from trace memory.

REPETITIVE EXECUTIONS WITH OVERVIEW DISPLAYS

There are two modes for making repetitive overview executions. These two modes are discussed separately in paragraphs a and b. The mode that you select will have an effect on the display of overview information.

a. **Overview Until User Halt Execute Mode.** This is the faster of the two repetitive modes. In the trace specification, select (overview) (until) (user_halt) and press (execute). Then select (show) (overview). In this mode, the state/software analyzer makes a continuous overview measurement. After the first complete filling of the overview-event memory, the measurement continues, overwriting the oldest event codes with new event codes. This measurement is never completed. The message "overview in process" will be shown on the STATUS line after the trace measurement is complete. Press the (halt) softkey and (RETURN) to end this repetitive mode.

b. **Overview Until Memory Full Execute Repeat Mode.** In the trace specification, select (overview) (until) (mem_full) and press (execute) (repeat). Then select (show) (overview). This mode makes a single measurement execution, filling both the trace memory and the overview-event memory. Then data acquisition ceases. Now the state/software analyzer updates its display. It holds its updated display momentarily, and then executes a new measurement. The repeating of these measurement executions will continue until you press the (halt) and (RETURN) keys.

NOTE

Repetitive modes are especially useful when taking histogram measurements with maximum-peak and floating-peak displays. Refer to the discussion of overview-event histograms for details.

OVERVIEW INFORMATION FROM OVERVIEW-EVENT MEMORY

The displays discussed in the remainder of this chapter require that a 20-channel acquisition board assembly be installed as POD 1 in your state/software analyzer. These displays are composed of event codes stored in the overview-event memory. From this information, the state/software analyzer can formulate histograms which show proportions of program time or activity distributed among your events, lists of event codes showing the order of occurrences of event-coded program routines, and graphs which show patterns of event occurrences.

HOW THE ANALYZER READS OVERVIEW-EVENT MEMORY

The information in the overview memory can be used to prepare histograms, lists, and graphs. Figure 11-2 shows how the overview memory is read for display preparation. The overview memory retains the order in which the event codes were captured, allowing preparation of the lists and graphs.

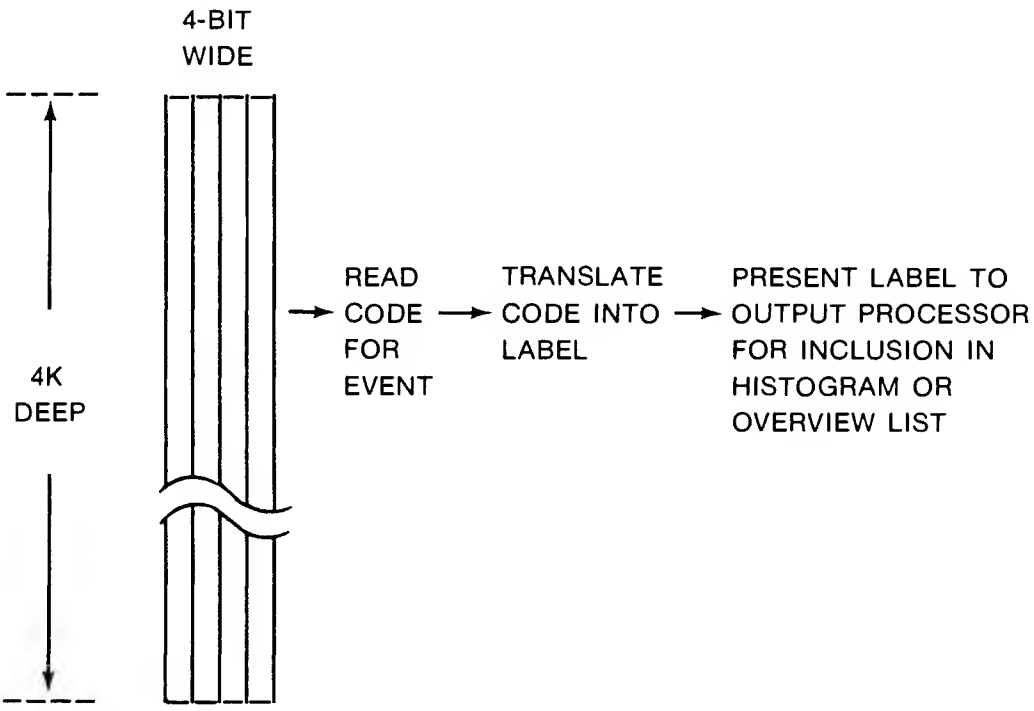


Figure 11-2. Reading The Overview-Event Memory

THE OVERVIEW-EVENT HISTOGRAM

The overview histogram shows the event names you assigned along with several representations of the activity observed in each event. The valid memory content is read to formulate the display, and then the following three types of information are shown on the display for each event:

1. The name of each event.

2. The absolute count of the number of times each event was stored in memory.

3. The percent of the measurement that met specifications for each of the events.

4. Inverse-video bars beside each event name. The length of each bar represents the portion of the measurement that met the specification of the associated event.

There are three modes of displaying an overview histogram: the default mode, the floating peak mode, and the maximum peak mode. These display modes operate as follows:

(display) (overview) (histogram) - This is the default mode of displaying an overview histogram. The horizontal bar graph is always scaled to show 100% of the overview event memory. The length of each bar on the graph is proportional to the 100% scaling of the display.

(display) (overview) (histogram) (max_peak) - This display is exactly the same as the default histogram display, except that the bar graph is rescaled. Each time the overview memory is filled, the display is expanded to bring the length of the longest bar to full-scale. The lengths of the other bars are adjusted in proportion to the rescaled display. The display scale is indicated by new percentage numbers which are shown across the top of the histogram.

This mode of display is most useful when you are making a repetitive series of overview measurement executions ((execute) (repeat) or (until) (user_halt)). In this mode, the display will be scaled to show the length of the longest bar ever detected during any measurement in the series. The name of the event that established the maximum scaling will be underlined.

Example: The scaling across the top may show a total display width of 59% of the memory space. The second event name on the display might be underlined. This indicates that during at least one of the measurements in the series of repetitive overview measurements, the second event occupied 59% of the space in the overview event memory, and this was the largest percentage of memory occupied by any single event during the series. (Event_02 may not be the biggest event on screen when you are looking at the display.)

(display) (overview) (histogram) (flt_peak) - This display is exactly the same as the display for the (max_peak) histogram, except that instead of retaining the maximum scaling for the entire series of repetitive measurements, the peak is only retained for five consecutive overview measurements. If a new greater peak is found during any measurement in the series, the display is immediately rescaled to reflect its value and that new scale is retained for five repetitive measurements. Then when the state/software analyzer has held a constant peak value for five repetitive measurements, it resets and starts a new series of overview measurements.

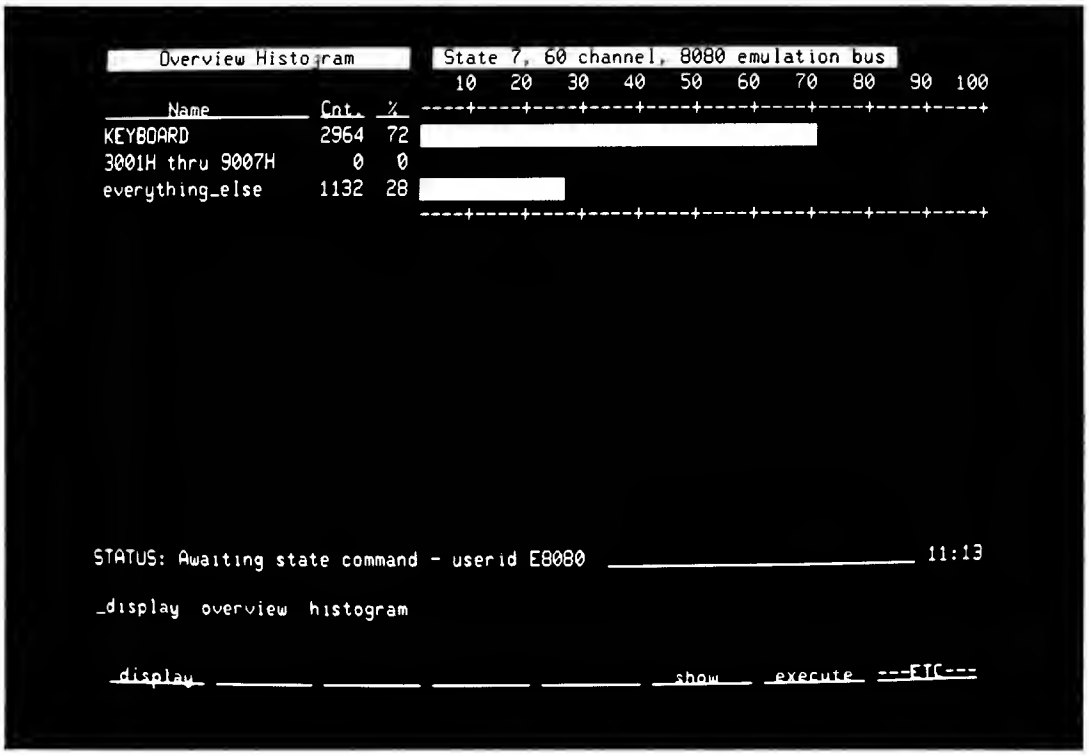


Figure 11-3. Typical Overview Histogram Display

THE OVERVIEW-EVENT LIST

The overview list display is a list of event names. The order of the names shows the order in which they occurred.

A summary is also presented on screen. It shows the total of each of the named events that were recorded during run execution. The summary includes the percentage of memory that contains each event.

This display can be used to determine the order of occurrence of program routines such as the order that a calling routine accesses several called routines. By defining events that are entry points to different routines, your list will show the order of calls made to those routines.

THE OVERVIEW-EVENT GRAPH

The overview graph shows your event names on the vertical axis and the order of occurrence of those events along the horizontal axis.

You can use this display to see the linkage between routines or program segments and observe patterns of activity. By adjusting the horizontal scale factor, you can obtain an overall view of all of the events in memory, and then focus on the details of activity around an area of interest. This information can be helpful when you are trying to recognize patterns of activity in your program routines and when you are characterizing the flow of your program processes.

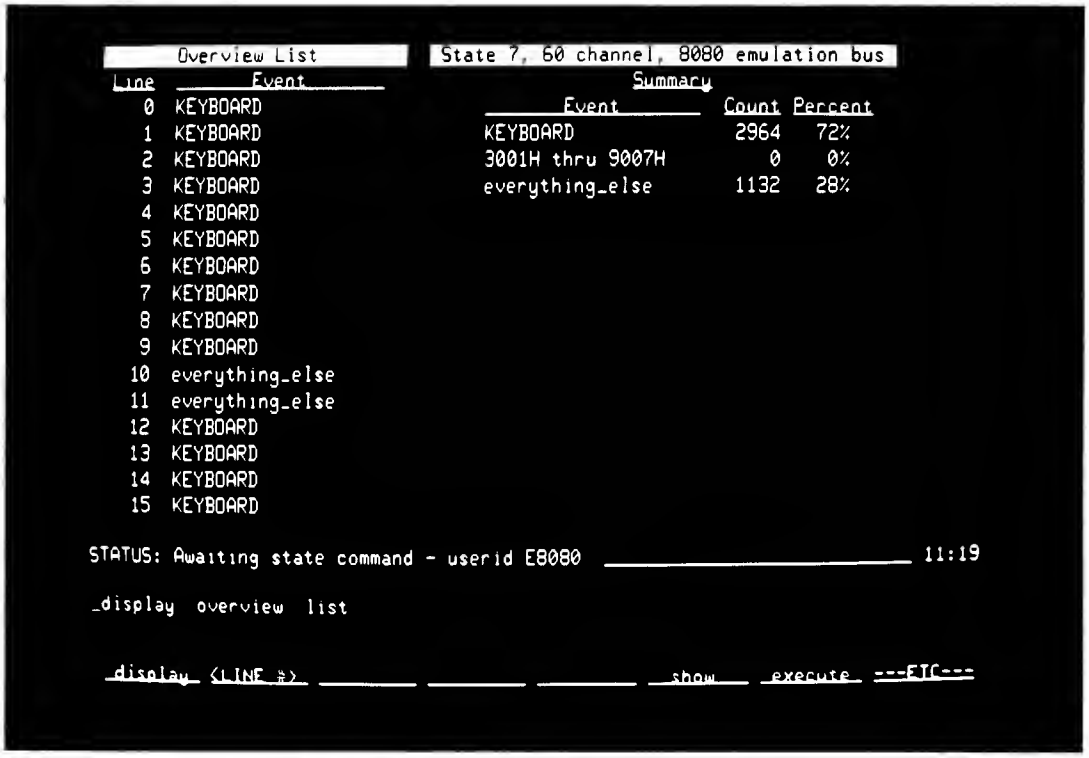


Figure 11-4. Typical Overview List Display

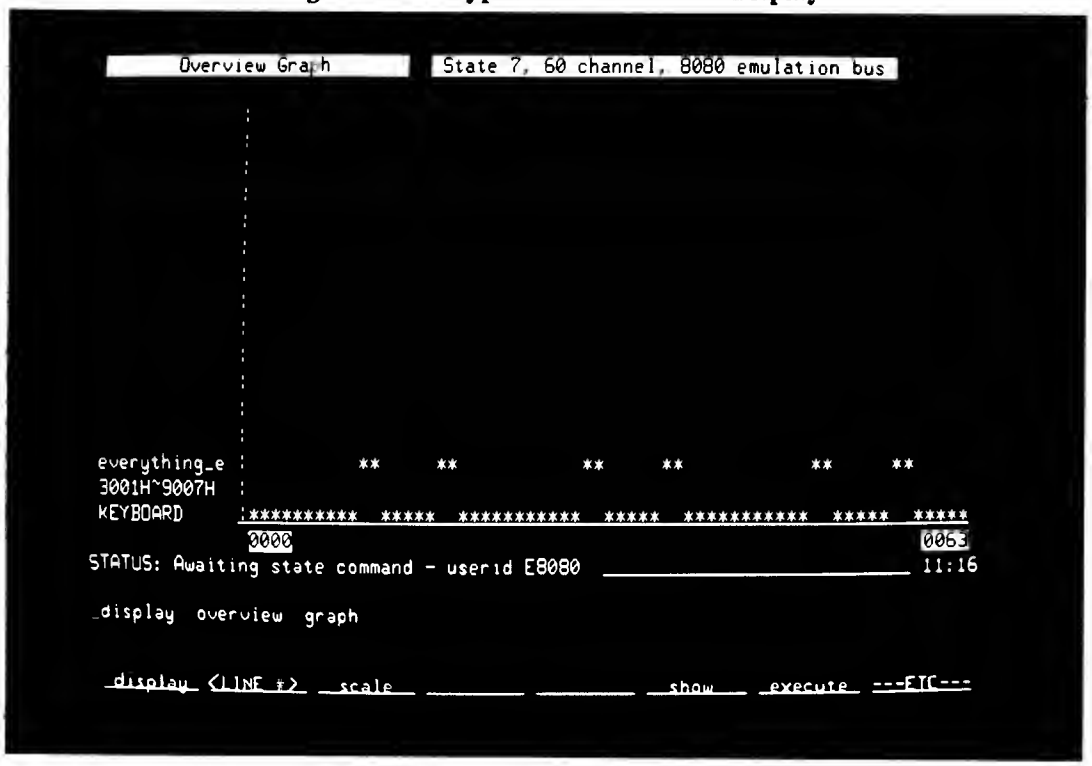


Figure 11-5. Typical Overview Graph Display

Appendix A
ERROR AND STATUS MESSAGES

MESSAGE DEFINITIONS

The following messages are displayed on screen to provide an indication of operating status. Some of these also advise you of improper operating conditions or invalid entries on the command line.

A map name may not also be a map symbol in another map - Displayed when you try to create a map of a name already assigned to represent a value or range on another map. The state/software analyzer will not accept a map and a symbol both with the same name.

A map symbol may not also be a map name - Displayed when you try to create a new symbol using a name that is already used as the name of a map. The state/software analyzer will not accept a map and a symbol both with the same name.

A symbol map must first be defined - Displayed when you enter a command to define a symbol and no maps have been defined in the map specification.

Absolute/database file name not assigned - Displayed when the state/software analyzer tries to satisfy a specification requiring assignment of an absolute/database file name when none has been assigned. You must assign a database file name before you can execute the present command on the command line.

Analyzer configuration does not include this pod - Displayed when you try to define a label in a pod that is not part of your state/software analyzer, such as labeling bits in pod 4 when your state/software analyzer has only three pods.

Analyzer load stopped - Displayed to indicate the analyzer was stopped during an analyzer load operation (after the "execute" key was pressed and before the measurement was begun).

Analyzer not loaded, measurement in process - Displayed only when two or more analyzers are installed in the same system. Normally, the system will load the analyzer hardware with the measurement configuration when you press "execute", and it will store the measurement configuration when you press "end". When another analyzer is executing a measurement, the circuitry assigned to perform this task is involved in making the measurement. The present measurement must be halted or completed before the circuitry will be available to load or store the configuration of the non-involved analyzer.

Appending sorted global symbols - Displayed when the state/software analyzer is reading all global symbols from the link_symbols file, sorting them, and appending them to the database.

Attempt to define too many labels - Displayed when the total of all labels, symbols, ranges, and values used in your state/software analyzer configuration exceeds the capacity of the label/symbol memory.

Attempt to define too many symbols - Displayed when the total of all the labels, symbols, ranges, and values used in your state/software analyzer configuration exceeds the capacity of the label/symbol memory.

At least one data label must be defined - Displayed when you try to exit the format specification when no data labels are defined for the incoming probe lines. You have to define at least one data label for the incoming probe lines in order to exit the format specification.

At least one sequence term must be defined - Displayed when you execute or save a configuration in which you have turned on the sequence element and deleted all sequence terms. The default sequence is "find any_state, enable"

Awaiting State Command - userid XXXXX - Displayed when the state/software analyzer is in a quiescent state, ready to accept a new command in its input command line.

Cannot change the absolute file with a measurement in process - Displayed when you try to specify a new absolute database file during execution of a measurement. You have to halt the measurement before you can specify a new absolute database file.

Cannot copy trace list with mnemonic debug - Displayed when you try to copy the trace list to a file or peripheral and you have specified mnemonic debug. The mnemonic debug capability uses the same space required by the copy command.

Checksum error, State analyzer is corrupted - Displayed when a file is loaded from memory and the checksum on the file does not agree with the checksum calculated by the state/software analyzer during the load operation.

Command requires absolute file name to be assigned - Displayed when you request "source on", "source only", "display line_numbers", "display ADDRESS relative_to segments", or "display ADDRESS relative_to symbols" if no absolute file name has been assigned. You must assign the name of the absolute file before your analyzer can obtain displays requested by any of these commands.

Command requires memory expander - Displayed when you request "source on", "source only", "display line_numbers", "display ADDRESS relative_to segments", or "display ADDRESS relative_to symbols" if no memory expander is installed with your analyzer. Your instrument must have a memory expander installed along with your analyzer in order to obtain the displays requested by any of these commands.

Configuring intermodule bus - Displayed when the state/software analyzer is loading a portion of a measurement configuration file that sets up parameters for making an interactive measurement.

Copying page number <NO.>, line number <NO.> - Displayed while the state/software analyzer is making a copy of the information you specified.

Copying trace list line +/-<NO.> - Displayed while the state/software analyzer is making a copy of the trace list memory content. The state/software analyzer updates this line during the copy process to show the progress of the operation.

Data label width must be <= 32 bits - Displayed when you try to define a label with too large a width. The state/software analyzer can process values up to 32 bits wide.

Data may not be saved into the file that is the data source - Displayed when you try to save your configuration "with data" into the file that you identified as the data source. The reason that the analyzer cannot perform this function is that while the acquisition memory is loaded from your configuration file, the data memory is not loaded. The only place where the data is available is the file memory. If you want to resave the data in a file, create a new file and copy it there.

Data may not be saved while a measurement is in process - Displayed when you try to save a measurement configuration with data while any analyzer in the system is performing a measurement. The circuitry required to save data in a file is used to perform the measurement. Either wait until the measurement is complete or halt the measurement. Then you can save your configuration with data.

Data pod bit must be between 0 and 19 - Displayed when you try to specify a bit number greater than 19. There are only 20 bits available in a pod.

Default file name not assigned - Displayed before any file name has been assigned in a specification. After a file name has once been assigned, it becomes the default file name for later specification entries.

Disc full - Displayed when you try to save a copy onto a disc memory that is already full.

Disc 0 is full - Displayed when you try to save a display or configuration on disc 0 when it has no storage space left.

Disc 0 is full, writing to disc 1 - Displayed when you save a file to a disc without specifying which disc to use. The state/software analyzer always tries to save to disc 0 first (if unspecified). If disc 0 is full, it automatically switches to disc 1 and saves the file there, if space is available.

Disc 1 is full - Displayed when you try to save a display or configuration on disc 1 when it has no storage space left.

Display memory overflow - Displayed when the number of measurement parameters you have entered exceeds the size of the display buffer. The state/software analyzer will still execute the measurement according to your parameters even though it cannot display those parameters for you.

Execution in process, command invalid - Displayed when the analyzer is executing a measurement and you try to change one of the specifications used in the measurement, such as to enter a new trigger specification.

Execution not allowed: incomplete intermodule bus config. - Displayed when operating in a multiple-module installation and you reset out of an analysis module. In this case, its configuration is not saved. When you try to execute a measurement at the meas_sys level, no configuration is available for the subject analyzer. You must return to the specifications of the analyzer and make the entries required, and then execute the measurement.

Expression too complex - Displayed when the analyzer finds an expression in a specification or on the command line that is too complicated to evaluate.

File exists, and is NOT a 64620 State Analysis file - Displayed when you try to load a configuration into the state/software analyzer from a file that is not a state/software analyzer configuration file (such as a timing analyzer configuration file).

File <filename> contains no line numbers - Displayed when you reference a line number in the source file and no line numbers are found, or the ASMB_sym file is not found when the analyzer builds the data base.

File <filename> is linked, but <data/prog/comn> segment invalid - Displayed when the state/software analyzer finds the file identified as the absolute file has one or more invalid segments after linking. When this occurs, relink the absolute file before proceeding.

File <filename> is linked, but only abs segments exist - Displayed when you try to use the range of an absolute, assembly-language file in trace specifications. The analyzer can use symbols defined in the file, but it cannot use ranging of any absolute segment because there can be many absolute segments in a file.

File <filename> is not linked - Displayed when you identify a file name as part of your linked absolute file and the state/software analyzer finds that the file has not been linked. Only linked files within your absolute file can be used by the state/software analyzer.

File is write protected - Displayed when you try to save a configuration under a file name that already contains a write-protected configuration.

File version <aabb> is incompatible with software version - Displayed when you try to load a configuration file into the analyzer and the analyzer finds that the configuration was created by a software version that is incompatible with the present version of software in your analyzer.

File was linked after assmb_db file was created - Displayed when analyzer finds the date on the present link_sym file is more recent then the date on the assembly database file. When this occurs, the analyzer will still allow use of the database, but it will show "(old data)" in the trace specification under the "ABSOLUTE" notation.

<filename> is too big - Displayed when you try to specify use of a disassembler whose size is greater than 2.5K words.

<filename> was not assembled by the IAL - Displayed when you enter a command to disassemble using a file that is not an inverse-assembler file.

"hpib" may only be used in stand alone mode - Displayed when you try to save a learn string to the HP-IB port when operating in a cluster configuration. The only time that the analyzer can generate a learn string to the HP-IB port is when it is operating in a stand-alone configuration, or it is being controlled by an external HP-IB controller.

IMB commands are invalid with only one analysis module - Displayed when you try to enter commands for an interactive measurement and the state/software analyzer is the only analysis module in the mainframe. Refer to the Interactive Measurements Chapter.

Internal only MASTER enable invalid with other IMB functions - Displayed when you try to save a configuration or execute a measurement involving interaction with other analyzer(s), and you specify the Master Enable to affect only the state/software analyzer (such as 'master enable on_window one). If you are going to activate master enable during an interactive measurement, it must be specified to interact with all analyzers in the measurement.

Internal TRIGGER enable invalid with overview trigger - Displayed when you try to enable your trigger search on a sequence or window element while triggering on an overview event. This is an invalid specification because Overview does not interact with the sequence and window elements.

Label is in use, see Trace List - Displayed when you try to delete a label that is defined as part of the display in the trace list.

Label is in use, see Trace Specification - Displayed when you try to delete a label that is used in one of the parameters in the trace specification.

Label used for mnemonic decode, see Trace List - Displayed when you try to delete a label that is used in parameters specified in the Trace Specification.

Line <line number> is not a range symbol - Displayed when you specify use of the range of code emitted by a line in the high-level source file, and the selected line does not emit multiple bytes/words of code. This happens on the last line of certain high-level source files.

Line <line number> is not found in file <filename> - Displayed when the state/software analyzer finds a specification referencing a line number in the high-level source file, but does not find any line having that number in the source file.

Loading analyzer - Displayed while the state/software analyzer is loading a measurement configuration into the acquisition hardware.

Loading linked symbols - Displayed when the state/software analyzer is preloading the memory expander with symbols from the specified database.

Loading State Analyzer with <filename> - Displayed while the state/software analyzer is loading a measurement configuration from a named file.

Map is in use, see Trace List - Displayed when you try to delete a map that is defined as part of the display in the trace list.

Map name already exists - Displayed when you attempt to rename a map to a name that has already been used to identify another map.

Map symbol already exists - Displayed when you attempt to rename a symbol to a name that has already been used by another symbol in the same map.

Measurement in process, may NOT corrupt specification - Displayed when you try to load a configuration from a file while a measurement is in process.

Multiple drivers - Displayed when you have assigned more than one analyzer to drive certain IMB lines. The only IMB line that can have more than one driver is the trigger line. Press the "end" softkey to obtain the meas_sys monitor and see which IMB line has too many drivers.

Multiple drivers on BNC PORT 1 - Displayed when you try to save a configuration for interactive measurement or execute an interactive measurement in which more than one of the analyzers was assigned to supply an output to BNC PORT 1 on the mainframe rear panel.

Multiple drivers on BNC PORT 2 - Displayed when you try to save a configuration for interactive measurement or execute an interactive measurement in which more than one analyzer was assigned to drive an output to BNC PORT 2 on the mainframe rear panel. BNC PORT 2 can only supply an output from one analyzer.

New linker, or new link_sym file format required - Displayed when the state/software analyzer finds your absolute file was linked using an old version of the linker that did not generate the new range records required to construct the database.

Not equal, "<>", may only be used in one type of sequence - Displayed when you try to overuse the "<>" entry in your sequence elements. You may use "<>" in sequence terms, or window one terms, or window two terms, or in restart terms, but only in one of these four places in a configuration.

No OVERVIEW events to display - Displayed when you enter a command to obtain an overview display following a measurement execution in which the overview function was turned on, but no overview events were defined.

(no link_sym file for time comparison) - Displayed when the state/software analyzer is unable to find and compare dates of the link_sym file and/or assembly database file to determine whether the database contains "old data".

Only one data label may use "range" - Displayed when you save or execute a measurement that specifies range measurements on two or more ranging labels. The state/software analyzer can only make ranging measurements on one ranging label at a time.

Only one function may drive IMB trigger enable - Displayed when you set up a specification in which both the trigger and the trigger-enable are specified to drive the IMB trigger enable line. You can only have one of these two functions drive the IMB trigger enable line in any configuration.

Only one OVERVIEW event may be everything_else - Displayed when you try to save a configuration or execute a measurement in which two overview events were assigned the range "else".

Only 15 display fields allowed - Displayed when your command line requires more than 15 columns of information. The state/software analyzer can format up to 15 columns of information.

Overlapping OVERVIEW ranges - Displayed when you try to save a measurement configuration or load it in the acquisition memory if it has overlapping ranges specified for certain of the Overview events. Refer to the Overview Measurements Chapter.

OVERVIEW COUNT invalid when doing overview on a data label - Displayed when you try to obtain an overview count of time or states during an "overview on label" measurement. The softkeys do not offer this mode, but you might accidently command it from a command file.

OVERVIEW COUNT ON invalid when doing overview on time count - Displayed when you try to enter a specification that requires an overview count of time and states in the same measurement. Refer to the Overview Measurement Chapter.

OVERVIEW displays invalid after a spec. change, EXECUTE again - Displayed when you have changed your overview measurement specification since making the last measurement, and you try to obtain a display of overview memory content.

OVERVIEW event is used for trigger - Displayed when you try to turn off the overview measurement system while the trace trigger function is using it for a trigger source.

OVERVIEW event range End_point less than Start_point - Displayed when you try to save a configuration or execute a measurement in which one or more of your overview events have ranges specified which end on a lower value than they start on. All ranges must end with values that are higher than their starting values.

OVERVIEW hardware is not present - Displayed when you enter a command that requires the overview hardware, and it is either missing or not installed as pod 1.

OVERVIEW interval may not be "always" - Displayed in the time count or state count mode of overview measurement if you have not specified points to begin and end each event count. In this case, the overview count is "always counting" which is a meaningless measurement.

OVERVIEW invalid, range is used - Displayed when your specification calls for use of a range and you also try to activate overview. You can use either ranges or the overview, but not both together.

OVERVIEW invalid, the overview is not on - Displayed when you try to request an overview graph, list, or histogram if the overview measurement function is off.

OVERVIEW not "on", command invalid - Displayed when you try to set up or delete overview parameters before you "turn on" the overview function. The softkeys are presented in an order to prevent this error, but the error may occur when you are entering your parameters from a command file instead of the softkeys.

Parse stack overflow - Displayed when the specification on the command line exceeds the space available in the state/software analyzer parse stack.

Premarking directory - Displayed when the state/software analyzer is reserving space in its file for index records. It reserves one index record space for each linked segment in the absolute database file.

Processing segment number 0000,FFFFFFFFF:UUUUUUU - Displayed when the state/software analyzer is separating all symbols and sorting them by segments and then adding the table of sorted symbols to the database.

Range End_point less than Start_point - Displayed when you execute or end a configuration in which one of your trace measurement specifications is for a range that begins with a higher value than its ending value. Ranges must begin with smaller numbers and end with larger numbers.

Range invalid, 'overview' in use - Displayed when your specification calls for use of overview and you also try to enter a trigger, store, or count specification requiring a range. You can use either ranges or overview, but not both together.

Reading index - Displayed when the state/software analyzer is reading the index to the specified database.

Request not possible - Displayed when you try to execute a measurement while a measurement is already in process, or try to halt a measurement while no measurements are in process.

Saving overview data in <filename> - Displayed while the state/software analyzer is saving the content of the overview memory into the named file.

Saving State Analyzer in <filename> - Displayed while the state/software analyzer is saving a measurement configuration into a named file.

Saving trace data in <filename> - Displayed while the state/software analyzer is saving the content of the data memory into the named file.

Scaling is invalid - Displayed when you try to enter a scaling command while you are in an Overview list display. No scaling is possible in a list display. Such an error can occur when operating from a command file.

Searching for segments in link_sym file - Displayed when the state/software analyzer is scanning the link_symbols file for segment records.

SEQUENCE cannot use "restart" with both WINDOWs on - Displayed when you try to save a configuration or load it in the acquisition memory if both window elements are turned on and a restart term is specified in the sequence element. Refer to the Sequence And Window Elements Chapter.

Stopped copying on line +/-<NO.> - Displayed if you stop a copy operation before the copy has been completed. The line number indicates how much of the trace list copy was completed before the copy was stopped.

Symbol is a don't care constant, and has no value - Displayed when you use a symbol that is defined as a don't care constant in any expression.

Symbol is in use, see Trace Specification - Displayed when you try to delete a symbol or map that is used by one of the parameters in the trace specification.

Symbol is not found in the specified map - Displayed when you try to enter a symbol and specify a map name as the location of the symbol, if that symbol is not defined there. Without specifying a map name, the analyzer will search the default map (if any), and then will assume the symbol is in the absolute database file, if specified.

Symbol overlaps with previously defined symbol - Displayed when you try to define a new symbol on a map and you assign a range that overlaps the range of another symbol on the map. No range can be represented by two symbols on the same map.

Symbol <symbol name> is not a range symbol - Displayed when you use a named symbol in a specification in a way requiring that symbol to be a ranging symbol, but that symbol identifies only a single value.

Symbol <symbol name> is not found in file <filename> - Displayed when the state/software analyzer cannot find a symbol that you identified as being resident to a particular file within your absolute file. Recheck the symbol name and the file specified when this is displayed.

Symbol <symbol name> is not found in globals - Displayed when the state/software analyzer finds a symbol identified as global in a specification, but does not find that symbol listed among the globals in the database.

Symbol value is identical to another symbol - Displayed when you try to enter a symbol with a value identical to the value of a symbol already defined for the present map. You may not have any value represented by two different symbols on the same map.

Syntax Error - Displayed when you try to enter an invalid command and press the RETURN key.

The MASTER ENABLE sequence may not be shared - Displayed when you try to save a configuration or execute a measurement in which the same sequence or window element is used by the master enable function and any other parameter. No other parameter can use the sequence or window element assigned to be used by the Master Enable function.

The OVERVIEW event is not defined - Displayed when you try to trigger, store, or count on the occurrence of an overview event that is undefined, or when you try to delete an overview event that is not defined.

The SEQUENCE is in use - Displayed when you try to turn off the SEQUENCE element while one of your measurement parameters is using the sequence element in its specification.

The SEQUENCE term is not defined - Displayed when you try to call a sequence term to the command line by the modify command, and the term you call is not defined.

The user may not scale the y axis - Displayed when the command input calls for scaling the y axis of the overview graph. This error can occur when operating from a command file input.

The WINDOW is in use - Displayed when you try to turn off a window element while one of your measurement parameters is using that window element in its specification.

The WINDOW is not defined - Displayed when you try to call the specification for a window element to the command line via the modify softkey, and that window is not defined.

Too many expressions - Displayed when the number of mathematical expressions in your present test specification exceeds the storage capacity of the state/software analyzer.

Too many "followed by" terms used - Displayed when you try to enter a sequence term that requires more sequence resources than are available to be used in any one term in the state/software analyzer. Refer to the Sequence And Window Chapter.

Too many "or" terms used - Displayed when you try to enter more OR terms in your trigger, store, and count specifications than the state/software analyzer can process. Refer to the Trace Measurements Chapter.

Too many OVERVIEW events specified - Displayed when you try to save a configuration or load it in the acquisition memory if it has more overview events than can be measured by the state/software analyzer. Refer to the Overview Measurements Chapter.

Too many ranges specified - Displayed when you have set up a trace specification requiring more resource patterns than are available in the state/software analyzer, and you try to save that configuration in a file. Refer to the Trace Measurements Chapter.

Too many resource patterns specified - Displayed when you have set up a trace specification requiring more resource patterns than are available in the state/software analyzer, and you try to save that configuration in a file or load it in the acquisition memory. Refer to the Trace Measurements Chapter.

Too many SEQUENCE terms required - Displayed when you try to enter a parameter that will use more sequence terms than available. Refer to the Sequence And Window Chapter.

Translating user's expressions - Displayed during normal loading of analyzer prior to an execution. This line advises that your expressions are being evaluated prior to loading them into the analyzer.

Trigger enable invalid with RECEIVED trigger - Displayed when your specification includes conditions on which to enable trigger search, and also specifies trigger to be received from another analyzer on the intermodule bus. The analyzer cannot operate on a specification for an internal trigger enable at the same time that it is set to receive trigger from an associated analyzer on the IMB.

Value is too large in an OVERVIEW event - Displayed when you try to save or load a configuration in which a count greater than the maximum count the state/software analyzer can process has been specified. The maximum time count that the state/software analyzer can process is 8.3 hours, and the maximum state count is 720 gigacounts.

Writing the directory - Displayed when the state/software analyzer is writing one index record for each linked segment, telling where in the database file the symbols for this segment exist.

-10.0 volts <= threshold <= 10.0 volts - Displayed when you try to establish a threshold voltage beyond the capabilities of the state/software analyzer threshold detection circuits.

STATUS LINE MESSAGES

LAST RUN MESSAGE

{single / Multiple} module execution {aborted / halted / complete} {Module not involved / Trace complete / Trace halted}

The state/software analyzer will use the above selection to formulate a statement that describes the status of the last measurement executed by the state/software analyzer.

RUN MESSAGE

{single / multiple} module execution {in process / repeating} {Module not involved / Slow clock / Waiting for trigger / Trace in process / Trace complete / Overview in process / Overview complete}

The state/software analyzer will use the above selections to formulate a statement describing the run it has in process.

Appendix B
GLOSSARY

This glossary contains all keywords, special symbols, and prompts which appear on the softkeys and in commands within the state/software analyzer. The following abbreviations indicate the locations where each of these appear:

FS - Format Specification
MS - Map Specification
TS - Trace Specification
PP - PreProcessor specification
LI - traceLIst
OV - OVerview displays

absolute
Indicate that the label should be interpreted without the use of a symbol map. Input and output are therefore absolute values.
FS - "define ADDRESS default_map absolute"
LI - "display ADDRESS absolute"

absolute_is absolute
Indicate the name of the absolute file to be traced and possible construction or updating of database for that file. Refer to Chapter 5.
FS MS TS PP LI OV - "absolute_is MONITOR"

activity_test activity
Used to add a line on the Format Specification display that shows the state of each bit in the analyzer.
FS - "activity_test"

address_is address
Used with "disassemble" to indicate the label that provides address information to the disassembler.
LI - "disassemble address_is ADDRESS"

after_term_number after
Used with "sequence insert" to place a new sequence term after an existing sequence term.
TS - "sequence insert after_term_number 1 find ADDRESS = 5500H"

all
Used with "copy tracelist" to indicate that the entire list be copied.
LI - "copy tracelist all to printer"

all_channels **all_chan**

Indicates that all clock pod input channels are to be affected.

FS - "threshold clock_pod all_channels ttl"

all_pods

Indicates that all clock and data pod input channels are to be affected.

FS - "threshold all_pods ttl"

all_specifications **all_specs**

Used with "copy" to copy the format, map, and trace specifications.

MS FS TS PP LI OV - "copy all_specifications to printer"

all_triggers **triggers**

Each time the trigger condition is found (not just the first time which is the trace point), a pulse will be sent on either the "stimulus_line", the "bnc_port_1", or both.

TS - "assert stimulus_line on all_triggers"

always

The indicated enable will always be true. This is used with trigger, store, count, overview, overview count, and master enable. When an enable is "always" true, it is not displayed in the trace specification (unless driving the IMB).

TS - "trigger enable always"

and

Used in two ways:

(1). To require that two or more fields (labels or clock inputs) match specified values to satisfy a condition.

FS - "clock is rising_edge channel_0 and high_level channel_1"

TS - "trigger on ADDRESS = 4000H and DATA = 0"

(2). In other tracing specifications to include two or more options.

TS - "store on sequence enable and disable"

any_state

Any state clocked into the analyzer will satisfy the function. Using "any_state" for trigger, store, count, or overview count does not require any state-recognition resources.

TS - "trigger on any_state"

any_value

Any value sampled for the label will be accepted for the condition. This is the same as a "don't care" specification for the label. Using "any_value" for trigger, store, count, or overview count requires a state-recognition resource.

TS - "trigger on ADDRESS = any_value"

append
Used with the "Copy" command to add information to the end of an existing file.
FS MS TS PP LI OV - "copy display to HISTORY append"

assert
Controls the drivers for the stimulus and halt functions to the preprocessor and the rear panel BNC port outputs. On/off, polarity, and signal source can be specified.
TS - "assert halt_line on measurement_complete"

as_entered as_enter
Used with "display" in the map specification to show each value in the base used when it was entered.
MS - "display as_entered"

before_term_number before
Used with "sequence insert" to place a new sequence term before an already existing sequence term.
TS - "sequence insert before_term_number 1 find ADDRESS = 5500H"

bnc_port_1 bnc_1
Refers to the BNC labeled `PORT 1' on the rear panel. This BNC can be used to output pulses (typically for stimulating the target system or triggering an oscilloscope) that occur on "all_triggers" or each time a specified point(s) in the sequence or a window is reached.
TS - assert bnc_port_1 on sequence enable"

bnc_port_2 bnc_2
Refers to the BNC labeled `PORT 2' on the rear panel. This BNC can be used to output a level (typically for halting the target system) that goes false at the start of a measurement and true at either "measurement_complete" or at "trace_point".
TS - assert bnc_port_2 on measurement_complete"

bnc_port_polarity polarity
Used with "assert" to specify the logic polarity to be output on rear panel `PORT 1' and `PORT 2'. Both outputs have the same polarity.
TS - "assert bnc_port_polarity positive"

both_edges both
Either edge on the indicated clock channel will clock the analyzer if the specified clock qualifiers are true on that edge.
FS - "clock_is both_edges channel_0"

calculate

Evaluate the arithmetic expression and display the result on the status line in the requested numerical base (default is hex). Calculations are performed with 32 bits and may use the following operators: + - * / & | mod. Parentheses may be used also. The order of precedence of operators (high to low) is:

```
- (unary)
mod  *  /
+  -
```

```
TS MS FS LI OV - "calculate in_dec 2+(3*4)"
```

center_of_trace center

The trace point (first state which satisfies the trigger condition) establishes the center of the trace list. The most recent 128 store-qualified states prior to the trace point are saved and an additional 127 store-qualified states are captured following the tracepoint.

```
TS - "trigger position_is center_of_trace"
```

channel_0

Refers to clock pod input channel 0 while specifying the edge or qualifier activity required on this channel.

```
FS - "clock_is rising_edge channel_0 and high_level channel_1"
```

channel_1

Refers to clock pod input channel 1 while specifying the edge or qualifier activity required on this channel.

```
FS - "clock_is rising_edge channel_1 and high_level channel_0"
```

channel_2

Refers to clock pod input channel 2 while specifying the edge or qualifier activity required on this channel.

```
FS - "clock_is rising_edge channel_2 and high_level channel_0"
```

channel_3

Refers to clock pod input channel 3 while specifying the edge or qualifier activity required on this channel.

```
FS - "clock_is rising_edge channel_3 and high_level channel_0"
```

channel_4

Refers to clock pod input channel 4 while specifying the edge or qualifier activity required on this channel.

```
FS - "clock_is rising_edge channel_4 and high_level channel_0"
```

channel_5

Refers to clock pod input channel 5 while specifying the edge or qualifier activity required on this channel.

```
FS - "clock_is rising_edge channel_5 and high_level channel_0"
```

channel_6
Refers to clock pod input channel 6 while specifying the edge or qualifier activity required on this channel.
FS - "clock_is rising_edge channel_6 and high_level channel_0"

channel_7
Refers to clock pod input channel 7 while specifying the edge or qualifier activity required on this channel.
FS - "clock_is rising_edge channel_7 and high_level channel_0"

channels_3_thru_0 ch_3_0
Indicates that only clock pod input channels 3 through 0 are to be affected.
FS - "threshold clock_pod channels_3_thru_0 ttl"

channels_7_thru_4 ch_7_4
Indicates that only clock pod input channels 7 through 4 are to be affected.
FS - "threshold clock_pod channels_7_thru_4 ttl"

clock_is clock
Specify the edges (and qualifiers) from the eight input channels on the clock pod to clock the analyzer. The edges are OR'd together and sample all qualifiers (if any are specified) together.
FS - "clock_is rising_edge channel_0 and low_level channel_1"

clock_pod
Used with "threshold" to indicate clock pod thresholds.
FS - "threshold clock_pod all_channels ttl"

comn
Used to specify the common segment of the absolute file.
TS - "trigger on ADDRESS = range file KEYBOARD comn"
MS - "define MAGIC range file MOVE comn"

configuration configure
Used to "load_from" or "save_in" a file. The file is type 'trace' and contains the entire configuration of the state/software analyzer.
MS FS TS PP LI OV - "configuration load_from SETUP:USER"

copy
Used to copy the "trace_specification", "format_specification", "map_specification", "all_specifications", or "display" to a listing file or to the printer. In addition, the "tracelist" can be copied from the trace list display.
MS FS TS PP LI OV - "copy display to printer"

count

Indicates that the trace count or overview count function is referenced. The trace count function controls increments to the counter whose value is stored with each store-qualified state in the trace list. The overview count function controls increments to the counter which is active during the interval in "overview on time_count" or "overview on state_count". Each counter increments only when both the "count enable" and the "count on" conditions are met.

```
TS - "count enable on window one"
TS - "count on time"
```

count_absolute **count_abs**

Includes absolute count (state or time as specified in the trace specification) as one of the columns of data in the trace list display). The zero point is at trigger and all other counts are with respect to this point. Negative counts indicate states captured prior to trigger.

```
LI - "display ADDRESS then mnemonic then count_absolute"
```

count_relative **count_rel**

Includes relative count (state or time as specified in the trace specification) as one of the columns of data in the trace list display. Each count value is referenced to the preceding state in the list. The first state always has a zero count.

```
LI - "display ADDRESS then mnemonic then count_relative"
```

counts

Used with "overview event" in "overview on state_count" mode to indicate that the unit for the number in the event range (or value) definition is single states.

```
TS - "overview event 1 is_the_range 0 counts thru 10 counts"
```

data

Used to specify the data segment of the absolute file.

```
TS - "trigger on ADDRESS = range file KEYBOARD data"
MS - "define MAGIC range file MOVE data"
```

data_is **data**

Used with "disassemble" to indicate the label that provides data information to the disassembler.

```
LI - "disassemble data_is DATA"
```

data_label **label**

Used with "display" to indicate that the identifier which follows is a label defined on one or more of the data pod input channels. The display will then show a summary of information for this label only.

```
FS - "display data_label ADDRESS"
```

data_pods
Used with "display" to display all labels defined on the data pods.
FS - "display data_pods"

debug
Used with "display mnemonic options" to produce a debug file showing the execution of the disassembler. This is useful to debug a user-defined disassembler. The output file is "TRC_DEBUG:HP:listing".
This option does not change the presentation on the display.
LI - "display mnemonic options debug"

decrement
Used with "scale" to reduce the value of one of the graph limits.
OV - "scale decrement x_axis lower_limit"

default
Used with "sequence", "window one", or "window two" to default the sequence or window to enable after any state is found. No other part of the trace specification is affected. The sequence or window is turned "on" if it was not already "on".
TS - "sequence default"

default_all default
Reloads the default configuration file, affecting all specifications and displays. The default file is determined by the type of probe or preprocessor attached to the analyzer.
FS - "default_all"

default_map def_map
Used with "define <LABEL>" to assign the default map to a label on the data input channels. Symbols in the default map will appear on the softkeys in the trace specification whenever the label is to be assigned a value (e.g. ADDRESS = INITIALIZE). The default map is also the default used in the trace list to interpret values sampled for the label.
FS - "define ADDRESS default_map ADDR_MAP"

default_trace_specification default
Changes only the trace specification to its simplest, or default, configuration: "trigger on any_state", "store on any_state", and "count on time". No sequencing or overview turned on.
TS - "default_trace_specification"

define

Used in two ways:
(1). Used to define new labels or modify the parameters of previously defined labels on the data input channels. Label position, "logic_polarity", and/or "default_map" may be specified.
FS - "define ADDRESS pod_1_bit 0 width 16"
(2). Used to define new symbols or modify the parameters of existing symbols in the map specification. Symbol values, and reference points for counting offset values in ranges can be defined using numbers, symbols already defined in a map, or symbols/segments defined in the absolute file under development.
MS - "define KEYBOARD range file KEYBOARD thru KEYBOARD end"

delay_clock **delay_clk**

Used with "master" to select to drive the analyzer's qualified clock over the Delay Clock function of the IMB.
TS - "master delay_clock driven_only"

delete

Remove the item named in this command from the specification.
FS - "delete ADDRESS"
TS - "sequence term_number 1 delete"
MS - "delete map"

disable

Used in two ways:
(1). With "sequence" to indicate that the sequence enable output should go false (disabled) after finding this term.
TS - "sequence term_number 1 find ADDRESS = 41ACH disable"
(2). To indicate that a function ("trigger", "store", or "assert") should be true on the disable state of the sequence or window.
TS - "trigger on sequence disable"

disable_after **disable**

Used with "window one" or "window two" to indicate that the window enable output should go false (disabled) after finding this term.
TS - "window one disable_after ADDRESS = 41DFH.

disabled

Used with "stimulus_line" to turn off the stimulus function of the preprocessor.
PP - "stimulus_line disabled"

disassemble **disassemb**

Used to indicate disassembly parameters for mnemonic decode. The disassembler, the starting line, the address label, the data label, and the status label to use for mnemonic decode can be specified.
LI - "disassemble address_is ADDRESS"

display

Used in three ways:
(1). To indicate which type of display is desired within a given display area.
MS - "display map ADDR_MAP"
FS - "display data_label ADDRESS"
LI - "display ADDRESS then mnemonic then count_relative"
OV - "display overview histogram"
(2). With "copy" to indicate that the display is to be copied to a file or to the printer.
MS FS TS PP LI OV - "copy display to printer"
(3). To create a new symbol map:
MS - "display map NEW_MAP"

div

Used the same as the slant bar "/" to indicate integer division.
TS FS MS LI OV - "calculate 10 div 5"

down

Used to roll the specified column down a selected number of spaces on the display.
LI - "display ADDRESS rolled down 5"

driven_only driven

Used with an IMB function to indicate that it is to be detected by the state/software analyzer and driven on the IMB to other modules, but not used internally to enable its own function. (The internal function is always enabled.)
TS - "trigger enable driven_only on_sequence"

drives_trigger_enable drive_te

Used with "trigger" to indicate that the state/software analyzer trigger function is to be driven to other modules over the IMB trigger enable signal. When IMB trigger enable is driven this way, it starts the measurement false and goes true when trigger is found.
TS - "trigger drives_trigger_enable on ADDRESS = 12AAH"

ecl

Used with "threshold" to indicate that an ECL level is desired. This is -1.3 volts.
FS - "threshold pod_1 ecl"

enable

Used in two ways:
(1). With "sequence" to indicate that the sequence enable output should go true (enabled) after finding this term.
TS - "sequence term_number 1 find ADDRESS = 41ACH enable"
(2). To indicate that a function ("trigger", "store", or "assert") should be true on the enable state of the sequence or window.
TS - "trigger on sequence enable"

enable_after — **enable**

Used with "window one" or "window two" to indicate that the window enable output should go true (enabled) after finding this term.

TS - "window one enable_after ADDRESS = 41ACH"

end

Used in three ways:

(1). Ends a state/software analysis session, saving the current configuration in a file, and returning to the level from which state/software analysis was called -- either the measurement_system or the monitor. The configuration file is named "SdcIJ:HP:trace" where: I = card slot of control board (0 to 8), and J = cluster address of station (0 to 7, or 8 if stand alone).

MS FS TS PP LI OV - "end"

(2). Used following a symbol that represents a range of values to indicate the value of the upper limit of the range.

MS TS FS LI OV - "calculate INITIALIZE map ADDR_MAP end"

TS - "trigger on ADDRESS = INITIALIZE end"

(3). Used with "copy tracelist thru" to request a list from the current position in the list thru the end of the list.

LI - "copy tracelist thru end to printer"

end_of_range — **end_range**

Used with "define <SYMBOL> range <I> thru <J> relative_to" in defining symbols within a map. Indicates that the values within the range are to be referenced to the end of the range for output (base value minus offset) and that the symbol itself has a value equal to the end of the range for input.

MS - "define INITIALIZE range 1432H thru 14A0H relative_to end_of_range"

end_of_trace — **end**

The trace point (first state which satisfies the trigger condition) establishes the end of the trace list. The most recent 255 store-qualified states prior to the trace point are saved.

TS - "trigger position_is end_of_trace"

enhanced

Used with "copy display" to indicate that each character in an enhanced video field on the display should be underscored in the output file or on the printer. (E.g. An `a' in inverse video would be output as 'a'.)

MS FS TS PP LI OV - "copy display enhanced to printer"

event

Used with "overview" to specify the name and range or value for each event to be detected by the overview measurement.

TS - "overview event 1 is_the_range 1432H thru 14A0H"

eventually eventual

Used with "occurring" in a sequence term to indicate that other states may occur while counting the occurrence of the specified sequence term.

```
TS - "sequence term_number 2 find ADDRESS = 49ACH occurring
eventually 5 times"
```

eventually_by eventual

Used with "followed" in a multiple term sequence line to indicate that the next term does not have to occur immediately after the preceding term.

```
TS - "sequence term_number 2 find ADDRESS = 1556H followed
eventually_by ADDRESS = 1600H
```

everything_else else

Used with "is_the_range" in specifying an overview event that represents all values not included in any other event.

```
TS - "overview event 5 is_the_range everything_else"
```

execute

Begins execution of a measurement. If interaction on the IMB is involved, all analyzers are started at the same time.

```
MS FS TS PP LI OV - "execute"
```

falling_edge falling

Used with "clock_is" to indicate that a falling edge (transition from high to low) on the indicated channel will clock the analyzer if the specified clock qualifiers are also true.

```
FS - "clock_is falling_edge channel_0"
```

file

Used to specify a source file within the absolute file.

```
TS - "trigger on ADDRESS = file T start"
MS - "define KEYBOARD range file KEYBOARD prog start
thru file KEYBOARD prog end"
```

find

Used to specify the conditions required to satisfy a sequence term.

```
TS - "sequence term_number 1 find ADDRESS = 1266H"
```

floating_peak flt_peak

Used with "display overview histogram" to control the scaling of the horizontal axis (occurrence) on the histogram. The horizontal scale will adjust to fit the largest histogram bar, increasing immediately and reducing after a short timeout. The event causing the largest scale is marked with an underscore.

```
OV - "display overview histogram floating_peak"
```

followed
Used in specifying a multiple term sequence line to indicate the next term in the line. Up to three "follow"s may be specified within each line (using four terms total from the sequencer).
TS - "sequence term_number 2 find ADDRESS = 1556H followed eventually_by ADDRESS = 1600H

format_specification **format**
Used with "show" to go to the format specification area of the state/software analyzer. The format specification contains:
Data pod label definitions
Clock pod edge and qualifier definitions
MS TS PP LI OV - "show format_specification"

from_line_number **from_line**
Used with "disassemble" to indicate on which line disassembly should start. Operation of this command is specific to each disassembler. (Typically the preprocessor provides enough status to the disassembler to make this command unnecessary.)
LI - "disassemble from_line_number 153"

giga_counts **giga_cts**
Used with "overview event" in "overview on state_count" mode to indicate that the unit for the number in the event range (or value) definition is one billion (1 000 000 000) counts.
TS - "overview event 1 is_the_range 1 giga_counts thru 10 giga_counts"

graph
Used with "display overview" to request a graph with each point on the y axis corresponding to one overview event and with the x axis corresponding to overview list line numbers. The x axis can be scaled.
OV - "display overview graph"

global
Used to identify a symbol as a global symbol in the absolute file.
TS - "trigger on ADDRESS = MOVE global"
MS - "define KEYBOARD value KEYBOARD global"

graph_of
Used with "display" to request a graph of the value of one data pod label as a function of trace list line number. The y axis corresponds to label value. The x axis corresponds to trace list line numbers. Both axes may be scaled.
OV - "display graph_of ADDRESS"

halt

Stops a measurement in process. If IMB interaction is involved, all analysis modules are halted. If the state/software analyzer had not yet found its trigger, a history of the most recent store-qualified states will be displayed.

```
MS FS TS PP LI OV - "halt"
```

halt_line

Refers to the halt signal sent to the preprocessor. This signal outputs a level (typically for halting the target system) that goes false at the start of a measurement and true at either "measurement_complete" or at "trace_point".

```
TS - "assert halt_line on measurement_complete"
```

handshaked handshake

Used to specify that the stimulus signal provided by the preprocessor will remain true until an acknowledge signal is returned to the preprocessor from the target system.

```
PP - "stimulus_line repetitive_mode handshaked"
```

high_level high

Used in specifying levels for the clock qualifiers in the "clock_is" command. The associated clock pod input channel must be sampled high by the clock edge to qualify the edge and clock the analyzer.

```
FS - "clock_is rising_edge channel_0 and high_level channel_1"
```

histogram

Used with "display overview" to request a histogram with one bar for each event defined in the trace specification. The length of each bar is proportional to the percentage of all events that were the event corresponding to the bar.

```
OV - "display overview histogram"
```

hpib

Used to specify the HP-IB output when operating in the stand-alone mode and performing a "Copy Tracelist".

```
TS MS FS LI OV - "copy tracelist to hpib"
```

immediately immediate

Used with "occurring" in a sequence term to indicate that the specified state must occur consecutively the specified number of times without any other state intervening. If a different state does occur, the count is reset and the search is resumed.

```
TS - "sequence term_number 2 find DATA = 0 occurring
immediately 5 times"
```

immediately_by immediate

Used with "followed" in a multiple term sequence line to indicate that the next term must occur immediately after the preceding term. If it does not, the search for the preceding term is resumed.

```
TS - "sequence term_number 2 find ADDRESS = 1556H followed
immediately_by ADDRESS = 1600
```

increment
Used with "scale" to increase the value of one of the graph limits.
OV - "scale increment x_axis lower_limit"

insert
Used with "sequence" to add a new term between or before previously defined terms.
TS - "sequence insert before_term_number 1 find ADDRESS = 5530H"

internal_and_driven int/drive
Used with an IMB function to indicate that it is to be detected by the state/software analyzer, driven on the IMB to other modules, and used internally to enable its own function.
TS - "trigger enable internal_and_driven on_sequence"

in_asc
Used with "display" to obtain values displayed in ASCII characters.
MS - "display in_asc"
LI - "display DATA in_asc"

in_bin
Used with "display" and "calculate" to specify binary as the numerical base for the value(s) displayed or the result to be calculated on the status line.
TS MS FS LI OV - "calculate in_bin 3ACH & 147H"

in_dec
Used with "display" and "calculate" to specify decimal as the numerical base for the value(s) displayed or the result to be calculated on the status line.
TS MS FS LI OV - "calculate in_dec 3ACH"

in_hex
Used with "display" and "calculate" to specify hexadecimal as the numerical base for the value to be displayed or the result to be calculated on the status line.
TS MS FS LI OV - "calculate in_hex 3ACH + 147H"

in_oct
Used with "display" and "calculate" to specify octal as the numerical base for the value to be displayed or the result to be calculated on the status line.
TS MS FS LI OV - "calculate in_oct 375Q + 147Q"

is_the_range is_range
Used with "overview event" to specify the range of values that will be decoded into this event.
TS - "overview event 1 is_the_range 3EDH thru 419H"

is_the_value is_value
Used with "overview event" to specify the single value that will be decoded into this event.
TS - "overview event 1 is_the_value 3EDH"

kilo_counts kilo_cts
Used with "overview event" in "overview on state_count" mode to indicate that the unit for the number in the event range (or value) definition is one thousand (1 000) counts.
TS - "overview event 1 is_the_range 1 kilo_counts thru 10 kilo_counts"

line
Used to identify a line number within a source file.
TS - "trigger on ADDRESS = line 5"
MS - "define NOT-JUMP value line 6"

line_numbers line_numb
Used to call a column of the line numbers from the source file to be shown on the trace list.
LI - "display line_numbers"

list
Used with "display overview" to request a listing of overview events in the order they were detected. The list is 4096 events long.
OV - "display overview list"

load_from
Used with "configuration" to configure the entire state/software analyzer as specified in a file. The file is of the type `trace'.
MS FS TS PP LI OV - "configuration load_from SETUP:USER"

logic_polarity polarity
Used with "define <LABEL>" to specify the logic polarity of the signals as probed. "Positive" logic polarity means that a voltage more positive than threshold will be interpreted as a `1'. "Negative" means it will be interpreted as a `0'.
FS - "define ADDRESS logic_polarity negative"

low_level low
Used in specifying levels for the clock qualifiers in the "clock_is" command. The associated clock pod input channel must be sampled low by the clock edge to qualify the edge and clock the analyzer.
FS - "clock_is rising_edge channel_0 and low_level channel_1"

lower_limit lower
Used with the "scale increment" or "scale decrement" commands to adjust the lower y-axis limit or the left x-axis limit.
OV - "scale increment x_axis lower_limit"

map

Used in three ways:
(1). Used with "delete" while displaying a map to delete any symbol on the map being displayed, or to delete the map being displayed along with all its symbols.
MS - "delete map"
(2). Used with "display" to call a map to the display.
MS - "display map ADDR_MAP"
(3). Used to identify the source of a symbol as a map.
TS - "trigger on ADDRESS = MOVE map ADDR_MAP"

map_specification **map_spec**

Used with "show" to go to the map specification area of the state/software analyzer.
TS PP LI OV FS - "show map_specification"

mapped

Used with "display mnemonic options" to request that address values be interpreted according to the default address map.
LI - "display mnemonic options mapped"

master

Refers to either "master enable" or "master delay_clock". The "master enable" function controls all state/software analysis activity. When this enable is false, all other functions of the state/software analyzer are frozen (including time count). The "master delay_clock" function is used to transfer qualified state clocks to other analysis modules over the IMB.
TS - "master enable on_sequence"
TS - "master delay_clock driven_only"

maximum_peak **max_peak**

Used with "display overview histogram" to control the scaling of the horizontal axis (occurrence) on the histogram. The horizontal scale will adjust to fit the largest histogram bar, increasing immediately and not reducing until a new measurement is begun. The event causing the largest scale is marked with an underscore.
OV - "display overview histogram maximum_peak"

measurement_complete **meas_comp**

Used with "assert bnc_port_2" or "assert halt_line". The BNC or halt_line will output a level (typically for halting the target system) that goes false at the start of a measurement and true at "measurement_complete", after trigger has been found and the trace memory has been filled.
TS - assert halt_line on measurement_complete"

mega_counts **mega_cts**

Used with "overview event" in "overview on state_count" mode to indicate that the unit for the number in the event range (or value) definition is one million (1 000 000) counts.
TS - "overview event 1 is_the_range 1 mega_counts thru 10 mega_counts"

memory_is_full **mem_full**

Used with "overview until" to capture only the first 4096 events into the overview memory. Subsequent events will not be stored, but will still be compared to the trigger event if "trigger on overview_event" was specified and was not yet found.

TS - "overview until memory_is_full"

min

Used with "overview event" in "overview on time_count" mode to indicate that the unit for the number in the event range definition is one minute.

TS - "overview event 1 is_the_range 1 min thru 10 min"

mnemonic

Includes mnemonic decode as one of the columns of data in the trace list display. A disassembler must be present, either by virtue of a configuration file, or via the "disassemble using" command.

LI - "display ADDRESS then mnemonic then count_absolute"

mod

One of the operators available in expressions. X mod Y produces the integral remainder left after dividing X by Y (0 <= X mod Y <= Y-1).

TS MS FS LI OV - "calculate 43 mod 5"
MS - "define START value 43 mod 5"
TS - "trigger on ADDRESS = 43 mod 5"

modify

Used to bring a specification down from the display onto the command line for modification and re-execution.

FS - "modify ADDRESS"
TS - "trigger modify"
LI - "display modify"

msec

Used with "overview event" in "overview on time_count" mode to indicate that the unit for the number in the event range definition is milliseconds.

TS - "overview event 1 is_the_range 1 msec thru 10 msec"

named

Used with "overview event" to specify a name to be used for the overview event in the overview histogram, list, and graph.

TS - "overview event 1 named START is_the_value 1000H"

negative

Used in two ways:
(1). Used with "define <LABEL>" to specify the logic polarity of the signals as probed is "negative". This means a voltage more negative than the channel threshold will be interpreted as a '1'.
FS - "define ADDRESS logic_polarity negative"
(2). Used with "assert bnc_port_polarity" to indicate the output polarity on the two rear panel BNCs. "Negative" indicates low pulses on `PORT 1' (if enabled) and a low level for halt on 'PORT 2' (if enabled).
TS - "assert bnc_port_polarity negative"

not_blocked **not_block**

Used to eliminate the block identifiers from columns displayed in the trace list.
LI - "display ADDRESS not_blocked"

non_inverse **non_inver**

Used to change the display of source file lines from inverse video to normal video.
LI - "source only non_inverse"

nothing

Used with "trigger on" to prevent any trigger from being recognized. This puts the trace memory into a mode similar to the "overview until user_halt" mode. The trace memory will always contain the most recent 256 store-qualified states and will display them as a history when the measurement is halted. This is useful to debug system-crash problems.
TS - "trigger on nothing"

occurring

Used when specifying sequence terms that must occur more than once before proceeding to the next term in the sequence. Occurrence on a multiple-term sequence line requires that the set of terms occurs in order the specified number of times.
TS - "sequence term_number 2 find ADDRESS = 499AH occurring eventually 5 times"

off

Used in two ways:
(1). Turns off the applicable tracing function. Used with "sequence", "window one", "window two", and "overview".
TS - "overview off"
(2). Turns off display of source file lines.
LI - "source off"

on

Used in three ways:
(1). To indicate the condition(s) that satisfy a function (as opposed to the enable of the function). This is used in "trigger", "store", "count", and "overview count".
TS - "trigger on ADDRESS = 1887H"
(2). To indicate the mode of overview operation.
TS - "overview on ADDRESS"
(3). To turn on display of source file lines in tracelist.
LI - "source on"

one

Used with "window" to indicate "window one" instead of "window two".
TS - "trigger enable on window one"

only

Used to turn on display of source file lines in tracelist without turning on display of any trace data.
LI - "source only"

on_sequence sequence

Used to specify the source of the enable for a function is the sequence. Used with master, trigger, store, count, and overview count enable.
TS - "master enable on_sequence"

on_window window

Used to specify the source of the enable for a function is window one or two. Used with master, trigger, store, count, and overview count enable.
TS - "master enable on_window one"

options

Used with "display mnemonic" to indicate options for the mnemonic display in the trace list. Either "mapped" or "debug" or both can be specified.
LI - "display mnemonic options mapped"

or

Used to indicate that any of two or more conditions can be true to satisfy the specification. When used with "trigger", "store", "count", or "overview count", each "or" requires another state-recognition resource. "Or" has lower priority than "and" in the trace specifications (i.e. A and B or C and D means (A and B) or (C and D)).
FS - "clock_is rising_edge channel_0 or falling_edge channel_1"
TS - "store on ADDRESS = 5000H or ADDRESS = 6000H"

or_less
Used with "overview event" in "overview on time_count" or "overview on state_count" modes to specify that all values less than or equal to the already specified value are included in the event.
TS - "overview event 1 is_the_range 1 msec or_less"

or_more
Used with "overview event" in "overview on time_count" or "overview on state_count" modes to specify that all values greater than or equal to the already specified value are included in the event.
TS - "overview event 1 is_the_range 1 sec or_more"

overview
Used in two ways:
(1). To control the overview measurement. The overview mode (address, state_count, or time_count), events, and enables can be specified.
TS - "overview on ADDRESS"
(2). To request one of the overview displays (histogram, list, or graph).
OV - "display overview histogram"

overview_displays overview
Used with "show" to go to the overview displays area of the state/software analyzer. The overview displays include:
Graphs for each label defined on the data pods
Overview histogram
Overview list
Overview graph
MS FS TS PP LI - "show overview_displays"

overview_event overview
Used with "trigger on" to request that the trace memory trace point be established by the first occurrence of a specific overview event.
TS - "trigger on overview_event 1"

pod_1
Used with "threshold" to indicate that the threshold for data pod 1 is to be adjusted.
FS - "threshold pod_1 ttl"

pod_1_bit
Used with "define <LABEL>" to indicate the starting (or ending) position of the label is contained within pod 1.
FS - "define ADDRESS pod_1_bit 0 thru pod_1_bit 15"

pod_2
Used with "threshold" to indicate that the threshold for data pod 2 is to be adjusted.
FS - "threshold pod_2 ttl"

pod_2_bit
Used with "define <LABEL>" to indicate the starting (or ending) position of the label is contained within pod 2.
FS - "define ADDRESS pod_2_bit 0 thru pod_2_bit 15"

pod_3
Used with "threshold" to indicate that the threshold for data pod 3 is to be adjusted.
FS - "threshold pod_3 ttl"

pod_3_bit
Used with "define <LABEL>" to indicate the starting (or ending) position of the label is contained within pod 3.
FS - "define ADDRESS pod_3_bit 0 thru pod_3_bit 15"

pod_4
Used with "threshold" to indicate that the threshold for data pod 4 is to be adjusted.
FS - "threshold pod_4 ttl"

pod_4_bit
Used with "define <LABEL>" to indicate the starting (or ending) position of the label is contained within pod 4.
FS - "define ADDRESS pod_4_bit 0 thru pod_4_bit 15"

pod_5
Used with "threshold" to indicate that the threshold for data pod 5 is to be adjusted.

pod_5_bit
Used with "define <LABEL>" to indicate the starting (or ending) position of the label is contained within pod 5.
FS - "define ADDRESS pod_5_bit 0 thru pod_5_bit 15"

pod_6
Used with "threshold" to indicate that the threshold for data pod 6 is to be adjusted.

pod_6_bit
Used with "define <LABEL>" to indicate the starting (or ending) position of the label is contained within pod 6.
FS - "define ADDRESS pod_6_bit 0 thru pod_6_bit 15"

position_is position
Used with "trigger" to position the trace point anywhere within the trace memory. The trigger position may be referenced from either the start or end of the trace memory. A position 1 after the start or 1 before the end places the trigger at the beginning or end, respectively.
TS - "trigger position_is 1 states_after_start"

positive

Used in two ways:
(1). Used with "define <LABEL>" to specify the logic polarity of the signals as probed is "positive". This means a voltage more positive than the channel threshold will be interpreted as a 1'.
FS - "define ADDRESS logic_polarity positive"
(2). Used with "assert bnc_port_polarity" to indicate the output polarity on the two rear panel BNCs. "Positive" indicates high pulses on `PORT 1' (if enabled) and a high level for halt on 'PORT 2' (if enabled).
TS - "assert bnc_port_polarity positive"

preprocessor_specification preproc

Used with "show" to go to the preprocessor specification of the state/software analyzer. This area is available only if a preprocessor is connected. The preprocessor specification controls the "stimulus_line" mode used in the preprocessor.
MS FS TS LI OV - "show preprocessor_specification"

printer

Used with "copy" to indicate that the requested information is to be listed on the system printer. Available only if the analyzer mainframe is connected to a cluster system with a printer.
MS FS TS PP LI OV - "copy display to printer"

prog

Used to specify the program segment of the absolute file.
TS - "trigger on ADDRESS = range file KEYBOARD prog"
MS - "define MAGIC range file MOVE prog"

pulsed

Used to specify that the stimulus signal provided by the preprocessor will pulse for approximately 5 microseconds.
PP - "stimulus_line single_mode pulsed"

range

Used in two ways:
(1). To indicate that a symbol represents a range of values.
MS - "define INITIALIZE range 1432H thru 14A0H"
(2). To specify that any value within a range of values satisfies a trace condition. This is used with "trigger on", "store on", and "count on".
TS - "trigger on ADDRESS = range INITIALIZE"

received

Used with an IMB function to indicate that it is to be received by the state/software analyzer as driven on the IMB by another module and used internally instead of its own function. This applies to "master enable", "trigger enable", "trigger", and "store enable".
TS - "trigger enable received"

received_or_driven rec/drive

Used with "trigger" when it interacts with the IMB to perform an 'OR' trigger. Two or more modules both receive trigger from the IMB and drive it with an internal function. The first such module to trigger will trigger all others receiving trigger from the IMB.

TS - "trigger received_or_driven on ADDRESS = 1000H"

relative_to relative

Used in two ways:

(1). To allow indication of the reference value to use when computing offsets for a symbol which represents a range of values.

MS - "define INITIALIZE range 1432H thru 14A0H relative_to start_of_range"

(2). To allow indication of which map to refer to when translating sampled values. The value is used to index into the symbol table to retrieve the symbol name or symbol plus offset to display in the list.

LI - "display ADDRESS relative_to ADDR_MAP"

rename

Used to change the name of a label, map, or symbol. Ability to rename any one depends upon the data displayed.

MS FS - "rename ADDRESS to ADDR_BUS"

repetitive_mode repeat

Used to specify that the stimulus signal provided by the preprocessor will be applied to the target system each time the specification is met as given in the trace specification "assert stimulus_line on" command.

PP - "stimulus_line repetitive_mode"

repetitively repeat

Used with "execute" to begin another measurement as soon as the current one is complete. Measurements will repeat until the "halt" command is issued.

MS FS TS PP LI OV - "execute repetitively"

restart_disable_on restart

Used when defining a sequence term that specifies a disable point to indicate under what condition the sequence should be restarted. If this condition is found before the disable point is reached, the sequence will restart to the first term in the disable subsequence (i.e. back to the first term after a previous enable or disable point).

TS - "sequence term_number 3 find ADDRESS = 1000H disable restart_disable_on ADDRESS = 7"

restart_enable_on restart

Used when defining a sequence term that specifies an enable point to indicate under what condition the sequence should be restarted. If this condition is found before the enable point is reached, the sequence will restart to the first term in the enable subsequence (i.e. back to the first term after a previous enable or disable point).

```
TS - "sequence term_number 3 find ADDRESS = 1000H enable
restart_enable_on ADDRESS = 7FF
```

rising_edge rising

Used with "clock_is" to indicate that a rising edge (transition from low to high) on the indicated channel will clock the analyzer if the specified clock qualifiers are also true.

```
FS - "clock_is rising_edge channel_0"
```

rolled

Used to offset the display of a column on a tracelist either up or down with respect to the other columns.

```
LI - "display ADDRESS rolled up 4"
```

save_in

Used with "configuration" to save the entire state/software analyzer configuration in a file. The file is of type `trace'.

```
MS FS TS PP LI OV - "configuration save_in SETUP:USER"
```

scale

Used to adjust the limits in a graph.

```
OV - "scale x_axis 9 to 100"
```

sec

Used with "overview event" in "overview on time_count" mode to indicate that the unit for the number in the event range definition is second(s).

```
TS - "overview event 1 is_the_range 1 sec thru 10 sec"
```

segments

Used to obtain displays of information from labels relative to segments within the absolute file. This selection excludes display of symbols.

```
LI - "display ADDRESS relative_to segments"
```

sequence

Used in two ways:

(1). To define sequence terms to build the SEQUENCE element of the trace specification.

```
TS - "sequence term_number 1 find ADDRESS = 69DBH"
```

(2). To assign the sequence enable output for use as a qualify condition for a function ("trigger", "store", and "assert").

```
TS - "trigger on sequence disable"
```

sequencer_status sequencer

Used with "display" to include the state of the sequence, window one, and window two as a column in the trace list. The sequence data shows current term number. All three show enable/disable status.

LI - "display sequencer_status"

show

Moves the user to another of the six display areas in the state/software analyzer. The six areas are:
"trace_specification"
"map_specification"
"format_specification"
"preprocessor_specification"
"tracelist"
"overview_displays"

FS MS TS PP LI OV - "show format_specification"

single_mode single

Used to specify that the stimulus signal provided by the preprocessor will only be applied to the target system the first time the specification is met as given in the trace specification "assert stimulus_line on" command.

PP - "stimulus_line single_mode"

source

Used to determine the content of a trace list display: either trace data only, source file statements only, or a combination of trace data and source statements.

LI - "source on"

space

Used to determine spacing between columns in tracelist displays.

LI - "display space 5"

start

Used in two ways:
(1). Following a symbol that represents a range of values to indicate use of the lower limit value of the range.

MS TS FS LI OV - "calculate INITIALIZE map ADDR-MAP start"

TS - "trigger on ADDRESS = INITIALIZE start"

(2). With "copy tracelist thru" to obtain a copy from the start of the list thru the current position.

LI - "copy tracelist thru start to printer"

start_of_range start_rng

Used with "define <SYMBOL> range <I> thru <J> relative_to" in defining symbols within a map. Indicates that offset values within the range are to be referenced to the start of the range for output (base value plus offset) and that the symbol itself has a value equal to the beginning of the range for input.

MS - "define INITIALIZE range 1432H thru 14A0H relative_to start_of_range"

start_of_trace start
The trace point (first state which satisfies the trigger condition) establishes the beginning of the trace list. The next 255 store-qualified states after the trace point are saved.
TS - "trigger position_is start_of_trace"

state_count state_cnt
Used with "overview on" to request the interval state count mode of operation. The interval is defined by a sequence or window in the "overview enable" command.
TS - "overview on state_count"

states_after_start after
Used with "trigger position_is <N>" to place trace point anywhere within the trace memory. The trigger position is referenced from the start of the trace memory. "Trigger position_is 1 states_after_start" is the same as "trigger position_is start_of_trace".
TS - "trigger position_is 10 states_after_start"

states_before_end before
Used with "trigger position_is <N>" to place trace point anywhere within the trace memory. The trigger position is referenced from the end of the trace memory. "Trigger position_is 1 states_before_end" is the same as "trigger position_is end_of_trace".
TS - "trigger position_is 10 states_before_end"

status_is status
Used with "disassemble" to indicate which label provides status information to the disassembler.
LI - "disassemble status_is STATUS"

stimulus_line stimulus
Used in two ways:
(1). To set the condition(s) upon which a pulse will be sent as the stimulus signal to the preprocessor.
TS - "assert stimulus_line on sequence enable"
(2). To control what the preprocessor will do with the stimulus signal when it is received. The preprocessor can be in either "single_mode" or "repetitive_mode" with either "pulsed" or "handshaked" operation.
PP - "stimulus_line single_mode pulsed"

store

Defines the requirements for the "store enable" and the "store on" specifications. "Store enable" can be "on_sequence", "on_window one", "on_window two", or "received" from the IMB. "Store on" can be from a sequence or window enable or disable or it can be one or more OR'd state-recognition resources. When both the "store enable" and the "store on" conditions are true, the state will be stored in the trace memory.

```
TS - "store enable on_sequence"
TS - "store on ADDRESS = 0A439H or ADDRESS = 0A43CH"
```

symbols

Used to obtain displays identifying label information in terms of symbols and segments from the absolute file.

```
LI - "display ADDRESS relative_to symbols"
```

term_number **term_num**

Used with "sequence" to define or modify one of the sequence terms.

```
TS - "sequence term_number 1 find ADDRESS = 4A36H"
```

terms_while_disabled **terms_dis**

Used with "trigger on" and "store on" to specify that the function is qualified on each state that also satisfies a term in the sequence while the sequence enable output is false (disabled).

```
TS "store on sequence terms_while_disabled"
```

terms_while_enabled **terms_en**

Used with "trigger on" and "store on" to specify that the function is qualified on each state that also satisfies a term in the sequence while the sequence-enable output is true (enabled).

```
TS "store on sequence terms_while_enabled"
```

then

Used with "display" to request two or more columns of data in the trace list.

```
LI - "display ADDRESS then mnemonic then count_relative"
```

threshold

Sets the threshold on a specific data pod (or group of channels in the clock pod) or on all pods. Thresholds can be "ttl", "ecl", or any voltage from "-10.0 volts" to "+10.0 volts".

```
FS - "threshold all_pods ttl"
```

thru

Used to indicate the second limit in a range of values.

```
MS - "define INITIALIZE range 1432H thru 14A0H"
TS - "overview event 1 is_the_range 1432H thru 14A0H"
LI - "copy tracelist thru 11 to printer"
```

time

Used with "count on" to request that the trace counter count time with 40-nS increments instead of counting states between each store-qualified state entered into the trace memory.

TS - "count on time"

time_count **time_cnt**

Used with "overview on" to request the interval time count mode of operation. The interval is defined by a sequence or window in the "overview enable" command.

TS - "overview on time_count"

times

Used with "occurring <N>" in a sequence term specification. Indicates the term must occur N times before the sequence can proceed.

TS - "sequence term_number 1 find ADDRESS = 8002H occurring 5 times"

to

Used with "copy" to indicate the destination for the listing.

MS FS TS PP LI OV - "copy display to printer"

trace_point **trace_pt**

Used with "assert bnc_port_2" or "assert halt_line". The BNC or halt_line will output a level (typically for halting the target system) that goes false at the start of a measurement and true at "trace_point", which is the point at which trigger specification is met the first time. This is the 'trigger' point as displayed in the trace list.

TS - assert halt_line on trace_point"

trace_specification **tracespec**

Used with "show" to go to the trace specification area of the state/software analyzer. The trace specification includes:
Trigger, store, count qualification for trace memory control.
Overview specification for overview memory control.
Sequence and window specification to enable other functions.
IMB interaction.
Master enable.
BNC, stimulus line, and halt line output control.

MS FS PP LI OV - "show trace_specification"

tracelist

Used with "show" to go to the trace list area of the state analyzer. The trace list displays the contents of the trace memory which includes:

Data as probed in the target system (displayed "absolute" or "relative_to" a map or to symbols/segments in the absolute-file) that met the store specification.

Counts (either state or time as specified in the trace specification) accumulated between each store qualified state

Mnemonic decode (disassembly) of processor instructions

High-level source-code statements

Sequence status

```
MS FS TS PP OV - "show tracelist"
```

trigger

Defines the requirements for the "trigger enable" and the "trigger on" specifications. "Trigger enable" can be "on_sequence", "on_window one", "on_window two", or "received" from the IMB. "Trigger on" can be from a sequence or window enable or disable, one or more OR'd state-recognition resources, or an "overview_event". When both the "trigger enable" and the "trigger on" conditions are true, the state will cause 'trigger'. The first such `trigger' is the trace memory trace point. All 'triggers' are output if "assert ... on all_triggers" is specified.

```
TS - "trigger enable on_sequence"
TS - "trigger on ADDRESS = 0A439H or ADDRESS = 0A43CH"
```

ttl

Used with "threshold" to indicate that a TTL level is desired. This is +1.4 volts.

```
FS - "threshold pod_1 ttl"
```

two

Used with "window" to indicate "window two" instead of "window one".

```
TS - "trigger enable on window two"
```

until

Used with "overview" to control the mode of the overview memory. The memory can accept events "until memory_is_full" or "until user_halt".

```
TS - "overview until memory_is_full"
```

up

Used with "rolled" to offset the position of a column in a trace list by a specified number of transactions.

```
LI - "display ADDRESS rolled up 5"
```

upper_limit upper

Used with the "scale increment" or "scale decrement" commands to adjust the upper y-axis limit or the right x-axis limit.

OV - "scale increment x_axis upper_limit"

usec

Used with "overview event" in "overview on time_count" mode to indicate that the unit for the number in the event range definition is microsecond(s).

TS - "overview event 1 is_the_range 1 usec thru 10 usec"

user_halt

Used with "overview until" to continuously capture events until the user halts the measurement. The last 4096 events are saved in the memory.

TS - "overview until user_halt"

using

Used with "disassemble" to load the disassembler for use in mnemonic decode. The disassembler is specified by name and is contained in a file: NAME:HP:reloc.

LI - "disassemble using I8085"

value

Used to indicate that a symbol represents a single value.

MS - "define START value 1000H"

volts

Used with "threshold" to specify a non-standard level to be used as the threshold for the pod(s) mentioned in the command. Standard thresholds are "ttl" (+1.4 volts) and "ecl" (-1.3 volts).

FS - "threshold pod_1 3.0 volts"

wait

Used to provide "wait" statements within command files to ensure completion of previous command statements, or to insert delays before command statements that change display contents.

FS MS TS - "wait 10"

width

Used in two ways:

(1). Used with "define <LABEL>" to indicate how many data pod input channels (bits) are assigned to the label. Bits are assigned starting with the lowest numbered bit and increasing (moving from right to left across the "data_pods" display). Labels can cross pod boundaries.

FS - "define ADDRESS pod_1_bit 0 width 24"

(2). Used in tracelist to specify the width of columns to be shown. This allows packing extra columns on a display.

LI - "display ADDRESS relative_to ADDR_MAP width 6"

window
Used in two ways:
(1). To define the conditions within `WINDOW ONE' and 'WINDOW TWO' that must be met to produce the enable and disable outputs from the windows.
TS - "window one enable_after ADDRESS = 69DBH disable_after ADDRESS = 710DH"
(2). To assign a window enable output for use as a qualify condition for a function ("trigger", "store", and "assert").
TS - "trigger on window one disable"

with_data
Used with the "configuration save_in <FILE>" command to save the present data memory content along with the measurement configuration.
MS TS FS LI OV - "configuration save_in KEEP with_data"

write_protect protect
Used with "configuration save_in <FILE>" to prevent the accidental modification of the file with a later "configuration save_in" command. The file is protected against writes only within the state/software analyzer and can still be purged, renamed, or copied into from the station monitor.
MS FS TS PP LI OV - "configuration save_in SETUP:USER write_protect"

x_axis
Used with "scale" to request an adjustment of the horizontal graph axis.
OV - "scale x_axis 0 to 100"

y_axis
Used with "scale" to request an adjustment of the vertical graph axis. The vertical axis in the overview graph cannot be adjusted.
OV - "scale y_axis 1000H to 1FFFH"

B
Suffixed to a number to indicate that its numerical base is binary (base 2). Each digit corresponds to one bit of the value.
MS FS TS LI OV - "0010000011110B"

D
Suffixed to a number to indicate that its numerical base is decimal (base 10). There is no direct correspondence between digits in the number and bits in value. Decimal numbers do not have to have a suffix.
MS FS TS LI OV - "1234D"

H

Suffixed to a number to indicate that its numerical base is hex (base 16). Each digit corresponds to four bits of the value. If the leading digit is A, B, C, D, E, or F, then the number must be prefixed with a 0.

```
MS FS TS LI OV - "0F12AH"
```

O

Suffixed to a number to indicate that its numerical base is octal (base 8). Each digit corresponds to three bits of the value.

```
MS FS TS LI OV - "173700"
```

Q

Suffixed to a number to indicate that its numerical base is octal (base 8). Each digit corresponds to three bits of the value.

```
MS FS TS LI OV - "173700"
```

X

Included within a number as a place holder to indicate that the digit may assume any value and still be accepted. "X" cannot be used with decimal numbers. The number of bits in the value represented by the "X" depends upon the numerical base of the number: binary - 1, octal - 3, hex - 4.

```
MS FS TS LI OV - "0010X000XXX10B"
```

: (colon)

Used in two ways:

(1). Used within file names to separate the userid and disc number fields from the file name.

```
MS FS TS PP LI OV - "copy display to FILE:USER:1"
```

(2). Used in place of the keyword "file" to specify the name of a source file within the linked absolute code under test. When used with a file name, it identifies the most recently named file. It can be used to override the default map for a label, if desired.

```
MS FS TS PP LI OV - "MOVE:MONITOR" specifies the MONITOR
  file as the location of the MOVE symbol.
```

? (question mark)

Used in place of the keyword "global" to designate a symbol as global to all source files linked within the absolute file under test.

```
TS - "trigger on ADDRESS = MOVE ?"
MS - "define KEYBOARD value KEYBOARD ?"
```

(pound sign)

Used in place of the keyword "line" to specify a line of code from a compiled source file.

```
MS TS FS PP LI OV - "#4 end" specifies the last addressable
  byte or word from the fourth line of the high-level
  source file).
```

; (semicolon)

Terminates a command on the command line. Text which follows a semicolon is ignored and may be used to comment a command file.

```
MS FS TS PP LI OV - "trigger on ADDRESS = 13A9H and
  DATA <> 0      :Look for RAM error"
```

((left parenthesis)

Used within expressions to group a set of operations together.

```
MS TS FS LI OV - "calculate (319H & 44H) + 1400H"
TS - "trigger on DATA = (5 & 7) +100H"
```

) (right parenthesis)

Used within expressions to group a set of operations together.

```
MS TS FS LI OV - "calculate (319H & 44H) + 1400H"
TS - "trigger on DATA = (5 & 7) +100H"
```

* (multiply)

Used within expressions to perform an arithmetic multiply on two 32-bit values.

```
MS TS FS LI OV - "calculate 149 * 330"
TS - "trigger on DATA = 149 * 330"
```

+ (plus)

Used in two ways:

(1). Used within expressions to perform an arithmetic addition on two 32-bit values.

```
MS TS FS LI OV - "calculate 319H + 44H"
TS - "trigger on DATA = 5 + 7"
```

(2). Used with threshold to indicate a positive voltage.

```
FS - "threshold pod_1 +5.0 volts"
```

- (minus)

Used in two ways:

(1). Used within expressions to perform an arithmetic subtraction on two 32-bit values.

```
MS TS FS LI OV - "calculate 319H - 44H"
TS - "trigger on DATA = 7 - 5"
```

(2). Used with threshold to indicate a negative voltage.

```
FS - "threshold pod_1 -1.8 volts"
```

/ (divide)

Used within expressions to perform an unsigned arithmetic integer divide on two 32-bit values.

```
MS TS FS LI OV - "calculate 319H / 44H"
TS - "trigger on DATA = 319H / 44H"
```

& (and)

Used within expressions to perform a logical AND on two 32-bit values.

```
MS TS FS LI OV - "calculate 319H & 44H"
TS - "trigger on DATA = 5 & 7"
```

| (or)
Used within expressions to perform a logical OR on two 32-bit values.
MS TS FS LI OV - "calculate 319H | 44H"
TS - "trigger on DATA = 5 | 7"

= (equals)
Indicates that the label must be detected with the specified value to satisfy the condition.
TS - "trigger on ADDRESS = 1449H"

<> (not equals)
Indicates that the label must be detected with other than the specified value to satisfy the condition.
TS - "trigger on ADDRESS = 1449H and DATA <> 0"

. (point)
Used to indicate the fractional part of a voltage.
FS - "threshold pod_1 -1.1 volts"

range Range invalid, 'overview' in use
TS - The same hardware is used to detect ranges for trigger, store, and count as is used to detect overview events. Both cannot be in use at the same time.

overview OVERVIEW invalid, no Overview hardware present
TS - The overview hardware is present on the 64623A board. This board must be part of the state/software analyzer and must be configured to receive data pod 1. If either condition is not satisfied, overview and ranging for trigger, store, and count are not available.

overview OVERVIEW invalid, 'range' is used
TS - The same hardware is used to detect ranges for trigger, store, and count as is used to detect overview events. Both cannot be in use at the same time.

overview OVERVIEW invalid, the overview is not on
OV - Overview displays cannot be shown unless a measurement has been run which included overview. Also, if any specification has been changed since the last measurement, no overview displays are available until the next measurement is started.

<ABBREVS> (file=:) (line=#) (global=?) (thru=~) (then=,)
FS MS TS LI OV - When this key is pressed, the STATUS line will define all of the abbreviations available to substitute for keywords in commands. These abbreviations can help you write shorter command lines.

<BIT #> Data bit number from 0 to 19
FS - The number of a data pod input channel.

<CMDFILE> Command file

FS MS TS LI OV - Any command file name can be entered here. It will execute the command file as soon as you press the (RETURN) key.

<DELAY> Time delay 0-65535

FS MS TS LI OV - Enter the desired delay in seconds for the wait command.

<EVENT #> Overview event number from 1 to 15

TS - At least 5 events can always be defined, plus "everything_else". Some definitions allow up to a total of 15 events.

<EXPRESS> Expressions may use: + - * / & | ( ) mod

MS FS TS LI OV - Any arithmetic expression can be used here. Calculations are performed with 32 bits. Results are truncated to label width if used to assign a value to a label in the trace specification or to a symbol in the map specification. Expressions may contain symbols as well as numerical values in binary, octal, decimal, or hex.

<FILE> Configuration file name

MS FS TS PP LI OV - The file will be of type `trace'. Names follow the standard 64000 file naming convention.

<FILE> Disassembler file name

LI - Just the name of the file. The state/software analyzer will supply the userid (HP) and file type (reloc).

<FILE> Listing file name

MS FS TS PP LI OV - The file will be of type `listing'. Names follow the standard 64000 file naming convention.

<HEADER> Listing header in quotes

FS TS MS LI OV - Enter any string to be placed at the top of any printed page.

<INVALID> Command syntax is invalid

MS FS TS PP LI OV - The portion of the command between the beginning of the command line and the cursor contains an error in syntax. Refer to the softkeys at each point in the command to verify syntax.

<LABEL> All data labels are specified

TS - The labels already used in the condition have specified values for at least one bit in all defined labels. (Only one of a set of overlapping labels can be used.)

<LABEL> No labels are defined

FS - The label definition area in the format specification is empty. Labels are created with the "define" command. If any labels are available, they will replace this prompt.

<LABEL> No ranging label defined

TS - "Range" or "overview on" only operate on labels which are entirely contained and right justified (start at bit 0) in pod 1 when pod 1 is connected to the 64623A board. The format specification "data_pods" display identifies the pod 1 connection with "ranging pod" on that pod. Also, the detailed display for each label ("display data_label <LABEL>") indicates "ranging_label" if that label can be used for ranging. If any ranging labels are available, they will replace this prompt.

<LINE #> Trace line number

LI OV - A line number present in the list, including negative values. A number out of range will be limited to the largest (or smallest) value.

<LINE> Source file line #

TS MS FS LI OV - A line number present in the source file.

<MAP> No symbol maps defined

LI - Maps are created in the map specification with the "display map <MAP>" command. If any maps are defined, their names will replace this prompt.

<MAP> Symbol map name

MS FS TS LI OV - The name of any map defined in the map specification may be entered.

<NAME> Name for the symbol of interest

MS - To create or rename a symbol, enter a unique name here.

<NAME> Name for new data label

FS - Enter a name for a new data label to be defined.

<NAME> No names defined

FS - This command cannot be performed unless a label (or symbol) is already defined. If any are defined, their names replace this prompt.

<NAME> Overview event name

TS - Each event can be given a name with up to nine characters.

<OFFSET> 1 <= Field offset <= 127

LI - Enter the desired offset for this column above or below the true trace line number.

<OCCUR> Sequence term occurrence from 1 to 65535

TS - Each sequence term can be required to repeat up to 65535 times before the sequence is advanced.

<PARMS> Parameters

TS FS MS LI OV - Enter parameters for command file.

<PATTERN> Value, up to 32 bits, may use 'X' (don't care)

MS TS - The value will be truncated to fit the width of the label. An `X' is a place holder which indicates that any value will be accepted in that position of the value. `X's are only valid with binary, octal, and hex numerical bases. Unspecified digits are assumed to be 0.

<RETURN> Command syntax is valid to cursor

MS FS TS PP LI OV - The portion of the command between the beginning of the command line and the cursor contains no errors in syntax and could be entered if no further options are desired.

<STATES> Trigger position in trace memory, 1 to 256

TS - The position is relative to either the start or end of memory as indicated by the next softkey in the command. A position of 1 is at either the beginning or end, exactly.

<STATES> Value from 0 to 720 giga counts

TS - State count values may be entered within this range. Resolution is a function of the value:

0 to	611 670 -	exact (1 in 1)
611 671 to	1 660 245 -	1 in 8
1 660 246 to	135 877 973 -	1 in 1024
135 877 974 to	17 315 747 157 -	1 in 131072
17 315 747 158 to	720 000 000 000 -	1 in 16777216

State count amounts up to 32 bits may be entered with units of "counts", "kilo_counts", "mega_counts", or "giga_counts".

<SYMBOL> No symbols in default map for label

TS - The default map for this label contains no symbols. Default_map is defined in the format specification. If any symbols are defined, they will replace this prompt.

<SYMBOL> Symbol map is empty

MS FS TS LI OV - This map contains no symbols. This is defined in the map specification. If any symbols are defined, they will replace this prompt.

<SYMBOL> Any map or linked symbol

MS - A map name can be any new name or any symbol defined in the absolute file. It cannot be the same as the name of a symbol already defined in a symbol map.

<SYMBOL> Any map symbol

FS TS MS LI OV - Enter any symbol defined in the default map.

<TERM #> Sequence term number from 1 to 3

TS - Only three sequence terms can be defined when both window one and window two are on. Also, no restart can be defined in this case. Each "followed" by in a sequence term also counts as a term.

<TERM #> Sequence term number from 1 to 7

TS - Only seven sequence terms can be defined when either window one or window two is on. Each "followed" by in a sequence term also counts as a term.

<TERM #> Sequence term number from 1 to 15

TS - Up to fifteen sequence terms can be defined if neither window one nor window two is on. Each "followed" by in a sequence term also counts as a term.

<TIME> Value from 0 to 480 minutes

TS - Time count values may be entered within this range. Resolution is a function of the value:

0	to	66 409 usec	-	40 nsec or 0/01%
66 msec	to	5 435 msec	-	40 usec or 0.01%
5 sec	to	693 sec	-	5 msec or 0.01%
11 min	to	480 min	-	670 msec or 0.01%

Time count amounts up to 32 bits may be entered with units of "usec", "msec", "sec", or "min".

<VALUE> Value, up to 32 bits

MS FS TS LI OV - The value will be maintained to 32-bit resolution throughout the evaluation of the expression. If the result is to be used with a label or symbol, it will be truncated to the width of the label or symbol in bits.

<VOLTAGE> Threshold voltage from -10.0 to +10.0 volts

FS - Pod thresholds may be set in 100-mV increments between +10.0 and -10.0 volts.

<WIDTH> Label width from 1 to 32 bits

FS - Labels are limited to 32 bits in width and must be contiguous (i.e. no bits may be excluded between the lowest and highest bit.) Labels may overlap each other, however.

<WIDTH> 1 <= Field width <= 63

LI - Each display field can be from 1 to 63 columns wide.

Appendix C

ASCII Conversion Table

This table lists the ASCII character set supported by the state/software analyzer. Non-standard ASCII characters were selected to identify trace data obtained during measurements.

ASCII Character	Bit Pattern 8765 4321	Dec Value	Hex Value	Octal Value
NU	000 0000	0	0	0
SH	000 0001	1	1	1
SX	000 0010	2	2	2
EX	000 0011	3	3	3
ET	000 0100	4	4	4
EQ	000 0101	5	5	5
AK	000 0110	6	6	6
BL	000 0111	7	7	7
BS	000 1000	8	8	10
HT	000 1001	9	9	11
LF	000 1010	10	A	12
VT	000 1011	11	B	13
FF	000 1100	12	C	14
CR	000 1101	13	D	15
SO	000 1110	14	E	16
SI	000 1111	15	F	17
DL	001 0000	16	10	20
D1	001 0001	17	11	21
D2	001 0010	18	12	22
D3	001 0011	19	13	23
D4	001 0100	20	14	24
NK	001 0101	21	15	25
ANY STRING	001 0110	22	16	26
EB	001 0111	23	17	27
ANY CHARACTER	001 1000	24	18	30
EM	001 1001	25	19	31
SB	001 1010	26	1A	32
EC	001 1011	27	1B	33
FS	001 1100	28	1C	34
GS	001 1101	29	1D	35
RS	001 1110	30	1E	36
US	001 1111	31	1F	37
SPACE	010 0000	32	20	40

ASCII Character	Bit Pattern 8765 4321	Dec Value	Hex Value	Octal Value
!	010 0001	33	21	41
"	010 0010	34	22	42
#	010 0011	35	23	43
$	010 0100	36	24	44
%	010 0101	37	25	45
&	010 0110	38	26	46
'	010 0111	39	27	47
(	010 1000	40	28	50
)	010 1001	41	29	51
*	010 1010	42	2A	52
+	010 1011	43	2B	53
,	010 1100	44	2C	54
-	010 1101	45	2D	55
.	010 1110	46	2E	56
/	010 1111	47	2F	57
0	011 0000	48	30	60
1	011 0001	49	31	61
2	011 0010	50	32	62
3	011 0011	51	33	63
4	011 0100	52	34	64
5	011 0101	53	35	65
6	011 0110	54	36	66
7	011 0111	55	37	67
8	011 1000	56	38	70
9	011 1001	57	39	71
:	011 1010	58	3A	72
;	011 1011	59	3B	73
<	011 1100	60	3C	74
=	011 1101	61	3D	75
>	011 1110	62	3E	76
?	011 1111	63	3F	77
@	100 0000	64	40	100
A	100 0001	65	41	101
B	100 0010	66	42	102
C	100 0011	67	43	103
D	100 0100	68	44	104
E	100 0101	69	45	105
F	100 0110	70	46	106
G	100 0111	71	47	107
H	100 1000	72	48	110
I	100 1001	73	49	111
J	100 1010	74	4A	112
K	100 1011	75	4B	113
L	100 1100	76	4C	114

ASCII Character	Bit Pattern 8765 4321	Dec Value	Hex Value	Octal Value
M	100 1101	77	4D	115
N	100 1110	78	4E	116
O	100 1111	79	4F	117
P	101 0000	80	50	120
Q	101 0001	81	51	121
R	101 0010	82	52	122
S	101 0011	83	53	123
T	101 0100	84	54	124
U	101 0101	85	55	125
V	101 0110	86	56	126
W	101 0111	87	57	127
X	101 1000	88	58	130
Y	101 1001	89	59	131
Z	101 1010	90	5A	132
[	101 1011	91	5B	133
\	101 1100	92	5C	134
]	101 1101	93	5D	135
^	101 1110	94	5E	136
_	101 1111	95	5F	137
`	110 0000	96	60	140
a	110 0001	97	61	141
b	110 0010	98	62	142
c	110 0011	99	63	143
d	110 0100	100	64	144
e	110 0101	101	65	145
f	110 0111	102	66	146
g	110 0111	103	67	147
h	110 1000	104	68	150
i	110 1001	105	69	151
j	110 1010	106	6A	152
k	110 1011	107	6B	153
l	110 1100	108	6C	154
m	110 1101	109	6D	155
n	110 1110	110	6E	156
o	110 1111	111	6F	157
p	111 0000	112	70	160
q	111 0001	113	71	161
r	111 0010	114	72	162
s	111 0011	115	73	163
t	111 0100	116	74	164
u	111 0101	117	75	165
v	111 0111	118	76	166
w	111 0111	119	77	167
x	111 1000	120	78	170
y	111 1001	121	79	171
z	111 1010	122	7A	172

ASCII Character	Bit Pattern 8765 4321	Dec Value	Hex Value	Octal Value
{	111 1011	123	7B	173
|	111 1100	124	7C	174
}	111 1101	125	7D	175
~	111 1110	126	7E	176
▨	111 1111	127	7F	177

INDEX

a

b

c

d

e

f

g

h

i

k

l

m

n

o

p

r

s

t

u

v

w

x

other

SALES & SUPPORT OFFICES

Arranged alphabetically by country

Product Line Sales/Support Key

Key Product Line
A Analytical
CM Components
C Computer Systems Sales only
CH Computer Systems Hardware Sales and Services
CS Computer Systems Software Sales and Services
E Electronic Instruments & Measurement Systems
M Medical Products
MP Medical Products Primary SRO
MS Medical Products Secondary SRO
P Personal Computation Products
*** Sales only for specific product line**
**** Support only for specific product line**

IMPORTANT: These symbols designate general product line capability. They do not insure sales or support availability for all products within a line, at all locations. Contact your local sales office for information regarding locations where HP support is available for specific products.

HP distributors are printed in italics.

HEADQUARTERS OFFICES

If there is no sales office listed for your area, contact one of these headquarters offices.

NORTH/CENTRAL AFRICA
Hewlett-Packard S.A.
7, Rue du Bois-du-Lan
CH-1217 **MEYRIN** 2, Switzerland
Tel: (022) 83 12 12
Telex: 27835 hpse
Cable: HEWPACKSA Geneve

ASIA
Hewlett-Packard Asia Ltd.
6th Floor, Sun Hung Kai Centre
30 Harbour Rd.
G.P.O. Box 795
HONG KONG
Tel: 5-832 3211
After Jan. 1, 1984
47th Floor, China Resources Bldg.
26 Harbour Rd., Wanchai
HONG KONG
Telex: 66678 HEWPA HX
Cable: HEWPACK HONG KONG

CANADA
Hewlett-Packard (Canada) Ltd.
6877 Goreway Drive
MISSISSAUGA, Ontario L4V 1M8
Tel: (416) 678-9430
Telex: 610-492-4246

EASTERN EUROPE
Hewlett-Packard Ges.m.b.h.
Lieblgasse 1
P.O.Box 72
A-1222 **VIENNA,** Austria
Tel: (222) 2365110
Telex: 1 3 4425 HEPA A

NORTHERN EUROPE
Hewlett-Packard S.A.
Uilenstede 475
P.O.Box 999
NL-1180 **AZ AMSTELVEEN**
The Netherlands
Tel: 20 437771

SOUTH EAST EUROPE
Hewlett-Packard S.A.
7, Rue du Bois-du-Lan
CH-1217 **MEYRIN** 2, Switzerland
Tel: (022) 83 12 12
Telex: 27835 hpse
Cable: HEWPACKSA Geneve

OTHER EUROPE
Hewlett-Packard S.A.
P.O. Box
150, Rte du Nant-O'Avril
CH-1217 **MEYRIN** 2, Switzerland
Tel: (022) 83 8111
Telex: 22486 hpsa
Cable: HEWPACKSA Geneve

MEDITERRANEAN AND MIDDLE EAST
Hewlett-Packard S.A.
Mediterranean and Middle East Operations
Atrina Centre
32 Kifissias Ave.
Paradissos-Amarousion, **ATHENS**
Greece
Tel: 682 88 11
Telex: 21-6588 HPAT GR
Cable: HEWPACKSA Athens

EASTERN USA
Hewlett-Packard Co.
4 Choke Cherry Road
ROCKVILLE, MD 20850
Tel: (301) 258-2000

MIDWESTERN USA
Hewlett-Packard Co.
5201 Tollview Drive
ROLLING MEADOWS, IL 60008
Tel: (312) 255-9800

SOUTHERN USA
Hewlett-Packard Co.
2000 South Park Place
P.O. Box 105005
ATLANTA, GA 30348
Tel: (404) 955-1500

WESTERN USA
Hewlett-Packard Co.
3939 Lankershim Blvd.
P.O. Box 3919
LOS ANGELES, CA 91604
Tel: (213) 506-3700

OTHER INTERNATIONAL AREAS
Hewlett-Packard Co.
Intercontinental Headquarters
3495 Deer Creek Road
PALO ALTO, CA 94304
Tel: (415) 857-1501
Telex: 034-8300
Cable: HEWPACK

ANGOLA
Telectra
Empresa Técnica de Equipamentos
R. Barbosa Rodrigues, 41-1 DT.
Caixa Postal 6487
LUANDA
Tel: 35515,35516
E,P

ARGENTINA
Hewlett-Packard Argentina S.A.
Avenida Santa Fe 2035
Martinez 1640 **BUENOS AIRES**
Tel: 798-5735, 792-1293
Telex: 17595 BIONAR
Cable: HEWPACKARG
A,E,CH,CS,P

Biotron S.A.C.I.M. e I.
Av Paseo Colon 221, Piso 9
*1399 **BUENOS AIRES***
Tel: 30-4846, 30-1851
Telex: 17595 BIONAR
M

AUSTRALIA

Adelaide, South Australia Office
Hewlett-Packard Australia Ltd.
153 Greenhill Road
PARKSIDE, S.A. 5063
Tel: 272-5911
Telex: 82536
Cable: HEWPARD Adelaide
A*,CH,CM,,E,MS,P

Brisbane, Queensland Office
Hewlett-Packard Australia Ltd.
10 Payne Road
THE GAP, Queensland 4061
Tel: 30-4133
Telex: 42133
Cable: HEWPARD Brisbane
A,CH,CM,E,M,P

Canberra, Australia Capital Territory Office
Hewlett-Packard Australia Ltd.
121 Wollongong Street
FYSHWICK, A.C.T. 2609
Tel: 80 4244
Telex: 62650
Cable: HEWPARD Canberra
CH,CM,E,P

Melbourne, Victoria Office
Hewlett-Packard Australia Ltd.
31-41 Joseph Street
BLACKBURN, Victoria 3130
Tel: 895-2895
Telex: 31-024
Cable: HEWPARD Melbourne
A,CH,CM,CS,E,MS,P

Perth, Western Australia Office
Hewlett-Packard Australia Ltd.
261 Stirling Highway
CLAREMONT, W.A. 6010
Tel: 383-2188
Telex: 93859
Cable: HEWPARD Perth
A,CH,CM,E,MS,P

Sydney, New South Wales Office
Hewlett-Packard Australia Ltd.
17-23 Talavera Road
P.O. Box 308
NORTH RYDE, N.S.W. 2113
Tel: 887-1611
Telex: 21561
Cable: HEWPARD Sydney
A,CH,CM,CS,E,MS,P

AUSTRIA
Hewlett-Packard Ges.m.b.h.
Grottenhofstrasse 94
A-8052 **GRAZ**
Tel: (0316) 291 5 66
Telex: 32375
CH,E

Hewlett-Packard Ges.m.b.h.
Lieblgasse 1
P.O. Box 72
A-1222 **VIENNA**
Tel: (0222) 23 65 11-0
Telex: 134425 HEPA A
A,CH,CM,CS,E,MS,P

BAHRAIN
Green Salon
P.O. Box 557
Manama
BAHRAIN
Tel: 255503-255950
Telex: 84419
P

Wael Pharmacy
P.O. Box 648
BAHRAIN
Tel: 256123
Telex: 8550 WAEL BN
E,C,M

BELGIUM
Hewlett-Packard Belgium S.A./N.V.
Blvd de la Woluwe, 100
Woluwedal
B-1200 **BRUSSELS**
Tel: (02) 762-32-00
Telex: 23-494 paloben bru
A,CH,CM,CS,E,MP,P

BRAZIL
Hewlett-Packard do Brasil I.e.C. Ltda.
Alameda Rio Negro, 750
Alphaville
06400 **BARUERI SP**
Tel: (011) 421.1311
Telex: (011) 33872 HPBR-BR
Cable: HEWPACK Sao Paulo
A,CH,CM,CS,E,M,P

Hewlett-Packard do Brasil I.e.C. Ltda.
Avenida Epitacio Pessoa, 4664
22471 **RIO DE JANEIRO-RJ**
Tel: (021) 286.0237
Telex: 021-21905 HPBR-BR
Cable: HEWPACK Rio de Janeiro
A,CH,CM,E,MS,P*

ANAMEO I.C.E.I. Ltda.
Rua Bage, 103
*04012 **SAO PAULO***
Tel: (011) 570-5726
Telex: 021-21905 HPBR-BR
M

SALES & SUPPORT OFFICES

Arranged alphabetically by country

CANADA

Alberta
Hewlett-Packard (Canada) Ltd.
3030 3rd Avenue N.E.
CALGARY, Alberta T2A 6T7
Tel: (403) 235-3100
A,CH,CM,E*,MS,P*

Hewlett-Packard (Canada) Ltd.
11120A-178th Street
EDMONTON, Alberta T5S 1P2
Tel: (403) 486-6666
A,CH,CM,CS,E,MS,P

British Columbia
Hewlett-Packard (Canada) Ltd.
10691 Shellbridge Way
RICHMOND,
British Columbia V6X 2W7
Tel: (604) 270-2277
Telex: 610-922-5059
A,CH,CM,CS,E*,MS,P*

Manitoba
Hewlett-Packard (Canada) Ltd.
380-550 Century Street
WINNIPEG, Manitoba R3H 0Y1
Tel: (204) 786-6701
A,CH,CM,E,MS,P*

Nova Scotia
Hewlett-Packard (Canada) Ltd.
P.O. Box 931
900 Windmill Road
DARTMOUTH, Nova Scotia B2Y 3Z6
Tel: (902) 469-7820
CH,CM,CS,E*,MS,P*

Ontario
Hewlett-Packard (Canada) Ltd.
3325 N. Service Rd., Unit 6
BURLINGTON, Ontario P3A 2A3
Tel: (416) 335-8644
CS,M*

Hewlett-Packard (Canada) Ltd.
552 Newbold Street
LONDON, Ontario N6E 2S5
Tel: (519) 686-9181
A,CH,CM,E*,MS,P*

Hewlett-Packard (Canada) Ltd.
6877 Goreway Drive
MISSISSAUGA, Ontario L4V 1M8
Tel: (416) 678-9430
A,CH,CM,CS,E,MP,P

Hewlett-Packard (Canada) Ltd.
2670 Queensview Dr.
OTTAWA, Ontario K2B 8K1
Tel: (613) 820-6483
A,CH,CM,CS,E*,MS,P*

Hewlett-Packard (Canada) Ltd.
220 Yorkland Blvd., Unit #11
WILLOWDALE, Ontario M2J 1R5
Tel: (416) 499-9333
CH

Quebec
Hewlett-Packard (Canada) Ltd.
17500 South Service Road
Trans-Canada Highway
KIRKLAND, Quebec H9J 2M5
Tel: (514) 697-4232
A,CH,CM,CS,E,MP,P*

Hewlett-Packard (Canada) Ltd.
Les Galeries du Vallon
2323 Ou Versont Nord
STE. FOY, Quebec G1N 4C2
Tel: (418) 687-4570
CH

CHILE

Jorge Calcagni y Cia. Ltda.
Av. Italia 634 Santiago
Casilla 16475
SANTIAGO 9
Tel: 222-0222
Telex: Public Booth 440001
A,CM,E,M

Olympia (Chile) Ltda.
Av. Rodrigo de Araya 1045
Casilla 256-V
SANTIAGO 21
Tel: (02) 22 55 044
Telex: 240-565 OLYMP CL
Cable: Olympiachile Santiagochile
CH,CS,P

CHINA, People's Republic of

China Hewlett-Packard Rep. Office
P.O. Box 418
1A Lane 2, Luchang St.
Beiwei Rd., Xuanwu District
BEIJING
Tel: 33-1947, 33-7426
Telex: 22601 CTSHP CN
Cable: 1920
A,CH,CM,CS,E,P

COLOMBIA

Instrumentación
H. A. Langebaek & Kier S.A.
Carrera 4A No. 52A-26
Apartado Aereo 6287
BOGOTA 1, D.E.
Tel: 212-1466
Telex: 44400 INST CO
Cable: AARIS Bogota
CM,E,M

Casa Humboldt Ltda.
Carrera 14, No. 98-60
Apartado Aereo 51283
BOGOTA 1, D.E.
Tel: 256-1686
Telex: 45403 CCAL CO.
A

COSTA RICA

Cientifica Costarricense S.A.
Avenida 2, Calle 5
San Pedro de Montes de Oca
Apartado 10159
SAN JOSE
Tel: 24-38-20, 24-08-19
Telex: 2367 GALGUR CR
CM,E,M

CYPRUS

Telerexa Ltd.
P.O. Box 4809
14C Stassinos Avenue
NICOSIA
Tel: 62698
Telex: 2894 LEVIDO CY
E,M,P

DENMARK

Hewlett-Packard A/S
Oatavej 52
OK-3460 **BIRKEROD**
Tel: (02) 81-66-40
Telex: 37409 hpas dk
A,CH,CM,CS,E,MS,P

Hewlett-Packard A/S
Rolighedsvej 32
OK-8240 **RISSKOV,** Aarhus
Tel: (06) 17-60-00
Telex: 37409 hpas dk
CH,E

DOMINICAN REPUBLIC

Microprog S.A.
Juan Tomás Mejía y Cotes No. 60
Arroyo Hondo
SANTO DOMINGO
Tel: 565-6268
Telex: 4510 ARENTA DR (RCA) P

ECUADOR

CYEDE Cia. Ltda.
Avenida Eloy Alfaro 1749
Casilla 6423 CCI
QUITO
Tel: 450-975, 243-052
Telex: 2548 CYEDE ED
CM,E,P

Hospitalar S.A.
Robles 625
Casilla 3590
QUITO
Tel: 545-250, 545-122
Telex: 2485 HOSPTL ED
Cable: HOSPITALAR-Quito
M

EGYPT

International Engineering Associates
24 Hussein Hegazi Street
Kasr-el-Aini
CAIRO
Tel: 23829, 21641
Telex: IEA UN 93830
CH,CS,E,M

EGYPOR
P.O.Box 2558
42 El Zahraa Street
CAIRO, Egypt
Tel: 65 00 21
Telex: 93 337
P

EL SALVADOR

IPESA de El Salvador S.A.
29 Avenida Norte 1216
SAN SALVADOR
Tel: 26-6858, 26-6868
Telex: 20539 IPESASAL
A,CH,CM,CS,E,P

FINLAND

Hewlett-Packard Oy
Revontulentie 7
PL 24
SF-02101 **ESPOO** 10
Tel: (90) 4550211
Telex: 121563 hewpa sf
CH,CM,CS,P

Hewlett-Packard Oy
(Olarinluoma 7)
PL 24
02101 **ESPOO** 10
Tel: (90) 4521022
A,E,MS

Hewlett-Packard Oy
Aatoksenkatv 10-C
SF-40720-72 **JYVASKYLA**
Tel: (941) 216318
CH

Hewlett-Packard Oy
Kainvuntie 1-C
SF-90140-14 **OULU**
Tel: (981) 338785
CH

FRANCE

Hewlett-Packard France
Z.I. Mercure B
Rue Berthelot
F-13763 Les Milles Cedex
AIX-EN-PROVENCE
Tel: 16 (42) 59-41-02
Telex: 410770F
A,CH,E,MS,P*

Hewlett-Packard France
64, rue Marchand Saillant
F-61000 **ALENCON**
Tel: 16 (33) 29 04 42

Hewlett-Packard France
Boite Postale 503
F-25026 **BESANCON**
28 rue de la Republique
F-25000 **BESANCON**
Tel: 16 (81) 83-16-22
CH,M

Hewlett-Packard France
13, Place Napoleon III
F-29000 **BREST**
Tel: 16 (98) 03-38-35

Hewlett-Packard France
Chemin des Mouilles
Boite Postale 162
F-69130 **ECULLY** Cedex (Lyon)
Tel: 16 (78) 833-81-25
Telex: 310617F
A,CH,CS,E,MP

Hewlett-Packard France
Tour Lorraine
Boulevard de France
F-91035 **EVRY** Cedex
Tel: 16 6 077-96-60
Telex: 692315F
E

Hewlett-Packard France
Parc d'Activite du Bois Briard
Ave. du Lac
F-91040 **EVRY** Cedex
Tel: 16 6 077-8383
Telex: 692315F
E

Hewlett-Packard France
5, avenue Raymond Chanas
F-38320 **EYBENS** (Grenoble)
Tel: 16 (76) 25-81-41
Telex: 980124 HP GRENOB EYBE
CH

Hewlett-Packard France
Centre d'Affaire Paris-Nord
Bâtiment Ampère 5 étage
Rue de la Commune de Paris
Boite Postale 300
F-93153 **LE BLANC MESNIL**
Tel: 16 (1) 865-44-52
Telex: 211032F
CH,CS,E,MS

Hewlett-Packard France
Parc d'Activités Cadera
Quartier Jean Mermoz
Avenue du Président JF Kennedy
F-33700 **MERIGNAC** (Bordeaux)
Tel: 16 (56) 34-00-84
Telex: 550105F
CH,E,MS

Hewlett-Packard France
Immueble "Les 3 B"
Nouveau Chemin de la Garde
ZAC de Bois Briand
F-44085 **NANTES** Cedex
Tel: 16 (40) 50-32-22
CH**

FRANCE (Cont'd)

Hewlett-Packard France
125, rue du Faubourg Bannier
F-45000 **ORLEANS**
Tel: 16 (38) 68 01 63

Hewlett-Packard France
Zone Industrielle de Courtaboeuf
Avenue des Tropiques
F-91947 Les Ulis Cedex **ORSAY**
Tel: (6) 907-78-25
Telex: 600048F
A,CH,CM,CS,E,MP,P

Hewlett-Packard France
Paris Porte-Maillot
15, Avenue de L'Amiral Bruix
F-75782 **PARIS** CEDEX 16
Tel: 16 (1) 502-12-20
Telex: 613663F
CH,MS,P

Hewlett-Packard France
124, Boulevard Tourasse
F-64000 **PAU**
Tel: 16 (59) 80 38 02

Hewlett-Packard France
2 Allée de la Bourgonnette
F-35100 **RENNES**
Tel: 16 (99) 51-42-44
Telex: 740912F
CH,CM,E,MS,P*

Hewlett-Packard France
98 Avenue de Bretagne
F-76100 **ROUEN**
Tel: 16 (35) 63-57-66
CH**,CS

Hewlett-Packard France
4 Rue Thomas Mann
Boite Postale 56
F-67033 **STRASBOURG** Cedex
Tel: 16 (88) 28-56-46
Telex: 890141F
CH,E,MS,P*

Hewlett-Packard France
Le Péripole
20, Chemin du Pigeonnier de la Cépière
F-31083 **TOULOUSE** Cedex
Tel: 16 (61) 40-11-12
Telex: 531639F
A,CH,CS,E,P*

Hewlett-Packard France
9, rue Baudin
F-26000 **VALENCE**
Tel: 16 (75) 42 76 16

Hewlett-Packard France
Carolor
ZAC de Bois Briand
F-57640 **VIGY** (Metz)
Tel: 16 (8) 771 20 22
CH

Hewlett-Packard France
Immeuble Péricentre
F-59658 **VILLENEUVE D'ASCQ** Cedex
Tel: 16 (20) 91-41-25
Telex: 160124F
CH,E,MS,P*

GERMAN FEDERAL REPUBLIC

Hewlett-Packard GmbH
Geschäftsstelle
Keithstrasse 2-4
D-1000 **BERLIN** 30
Tel: (030) 24-90-86
Telex: 018 3405 hpbln d
A,CH,E,M,P

Hewlett-Packard GmbH
Geschäftsstelle
Herrenberger Strasse 130
D-7030 **BÖBLINGEN**
Tel: (7031) 14-0
Telex:
A,CH,CM,CS,E,MP,P

Hewlett-Packard GmbH
Geschäftsstelle
Emanuel-Leutze-Strasse 1
D-4000 **DUSSELDORF**
Tel: (0211) 5971-1
Telex: 085/86 533 hpdd d
A,CH,CS,E,MS,P

Hewlett-Packard GmbH
Geschäftsstelle
Schleefstr. 28a
D-4600 **DORTMUND**-Aplerbeck
Tel: (0231) 45001

Hewlett-Packard GmbH
Vertriebszentrale Frankfurt
Berner Strasse 117
Postfach 560 140
D-6000 **FRANKFURT** 56
Tel: (0611) 50-04-1
Telex: 04 13249 hpffm d
A,CH,CM,CS,E,MP,P

Hewlett-Packard GmbH
Geschäftsstelle
Aussenstelle Bad Homburg
Louisenstrasse 115
D-6380 **BAD HOMBURG**
Tel: (06172) 109-0

Hewlett-Packard GmbH
Geschäftsstelle
Kapstadtring 5
D-2000 **HAMBURG** 60
Tel: (040) 63804-1
Telex: 021 63 032 hphh d
A,CH,CS,E,MS,P

Hewlett-Packard GmbH
Geschäftsstelle
Heidering 37-39
D-3000 **HANNOVER** 61
Tel: (0511) 5706-0
Telex: 092 3259
A,CH,CM,E,MS,P

Hewlett-Packard GmbH
Geschäftsstelle
Rosslauer Weg 2-4
D-6800 **MANNHEIM**
Tel: (0621) 70050
Telex: 0462105
A,C,E

Hewlett-Packard GmbH
Geschäftsstelle
Messerschmittstrasse 7
D-7910 **NEU ULM**
Tel: 0731-70241
Telex: 0712816 HP ULM-D
A,C,E*

Hewlett-Packard GmbH
Geschäftsstelle
Ehhericherstr. 13
D-8500 **NÜRNBERG** 10
Tel: (0911) 5205-0
Telex: 0623 860
CH,CM,E,MS,P

Hewlett-Packard GmbH
Geschäftsstelle
Eschenstrasse 5
D-8028 **TAUFKIRCHEN**
Tel: (089) 6117-1
Telex: 0524985
A,CH,CM,E,MS,P

GREAT BRITAIN

See United Kingdom

GREECE

Kostas Karaynnis S.A.
8 Omirou Street
***ATHENS** 133*
Tel: 32 30 303, 32 37 371
Telex: 215962 RKAR GR
A,CH,CM,CS,E,M,P

PLAISIO S.A.
G. Gerardos
24 Stournara Street
ATHENS
Tel: 36-11-160
Telex: 221871
P

GUATEMALA

IPESA
Avenida Reforma 3-48, Zona 9
GUATEMALA CITY
Tel: 316627, 314786
Telex: 4192 TELTRO GU
A,CH,CM,CS,E,M,P

HONG KONG

Hewlett-Packard Hong Kong, Ltd.
G.P.O. Box 795
5th Floor, Sun Hung Kai Centre
30 Harbour Road
HONG KONG
Tel: 5-8323211
Telex: 66678 HEWPA HX
Cable: HEWPACK HONG KONG
E,CH,CS,P

CET Ltd.
1402 Tung Wah Mansion
199-203 Hennessy Rd.
*Wanchia, **HONG KONG***
Tel: 5-729376
Telex: 85148 CET HX
CM

Schmidt & Co. (Hong Kong) Ltd.
Wing On Centre, 28th Floor
Connaught Road, C.
HONG KONG
Tel: 5-455644
Telex: 74766 SCHMX HX
A,M

ICELAND

Elding Trading Company Inc.
Hafnarnvoli-Tryggvagotu
P.O. Box 895
*IS-**REYKJAVIK***
Tel: 1-58-20, 1-63-03
M

INDIA

Computer products are sold through Blue Star Ltd. All computer repairs and maintenance service is done through Computer Maintenance Corp.

Blue Star Ltd.
Sabri Comptex II Floor
24 Residency Rd.
***BANGALORE** 560 025*
Tel: 55660
Telex: 0845-430
Cable: BLUESTAR
A,CH,CM,CS*,E*

Blue Star Ltd.
Band Box House
Prabhadevi
***BOMBAY** 400 025*
Tel: 422-3101
Telex: 011-3751
Cable: BLUESTAR
A,M

Blue Star Ltd.
Sahas
414/2 Vir Savarkar Marg
Prabhadevi
***BOMBAY** 400 025*
Tel: 422-6155
Telex: 011-4093
Cable: FROSTBLUE
A,CH,CM,CS*,E,M*

Blue Star Ltd.
Kalyan, 19 Vishwas Colony
*Alkapuri, **BORODA**, 390 005*
Tel: 65235
Cable: BLUE STAR
A

Blue Star Ltd.
7 Hare Street
***CALCUTTA** 700 001*
Tel: 12-01-31
Telex: 021-7655
Cable: BLUESTAR
A,M

Blue Star Ltd.
133 Kodambakkam High Road
***MADRAS** 600 034*
Tel: 82057
Telex: 041-379
Cable: BLUESTAR
A,M

Blue Star Ltd.
Bhandari House, 7th/8th Floors
91 Nehru Place
***NEW DELHI** 110 024*
Tel: 682547
Telex: 031-2463
Cable: BLUESTAR
A,CH,CM,CS*,E,M*

Blue Star Ltd.
15/16:C Weltestey Rd.
***PUNE** 411 011*
Tel: 22775
Cable: BLUE STAR
A

Blue Star Ltd.
2-2-47/1108 Bolarum Rd.
***SECUNDERABAD** 500 003*
Tel: 72057
Telex: 0155-459
Cable: BLUEFROST
A,E

Blue Star Ltd.
T.C. 7/603 Poornima
Maruthankuzhi
***TRIVANDRUM** 695 013*
Tel: 65799
Telex: 0884-259
Cable: BLUESTAR
E

Computer Maintenance Corporation Ltd.
115, Sarojini Devi Road
***SECUNDERABAD** 500 003*
Tel: 310-184, 345-774
Telex: 031-2960
*CH***

SALES & SUPPORT OFFICES

Arranged alphabetically by country

INDONESIA

BERCA Indonesia P.T.
P.O.Box 496/Jkt.
Jl. Abdul Muis 62
JAKARTA
Tel: 21-373009
Telex: 46748 BERSAL IA
Cable: BERSAL JAKARTA
P

BERCA Indonesia P.T.
P.O.Box 2497/Jkt
Antara Bldg., 17th Floor
Jl. Medan Merdeka Selatan 17
JAKARTA-PUSAT
Tel: 21-344-181
Telex: BERSAL IA
A,CS,E,M

BERCA Indonesia P.T.
P.O. Box 174/SBY.
Jl. Kutei No. 11
SURABAYA
Tel: 68172
Telex: 31146 BERSAL SB
Cable: BERSAL-SURABAYA
A,E,M,P*

IRAQ

Hewlett-Packard Trading S.A.
Service Operation
Al Mansoor City 9B/3/7
BAGHDAD
Tel: 551-49-73
Telex: 212-455 HEPAIRAQ IK
CH,CS

IRELAND

Hewlett-Packard Ireland Ltd.
82/83 Lower Leeson Street
DUBLIN 2
Tel: 0001 608800
Telex: 30439
A,CH,CM,CS,E,M,P

Cardiac Services Ltd.
Kilmore Road
Artane
DUBLIN 5
Tel: (01) 351820
Telex: 30439
M

ISRAEL

Eldan Electronic Instrument Ltd.
P.O.Box 1270
JERUSALEM 91000
16, Ohaliav St.
JERUSALEM 94467
Tel: 533 221, 553 242
Telex: 25231 AB/PAKRD IL
A

Electronics Engineering Division
Motorola Israel Ltd.
16 Kremenetski Street
P.O. Box 25016
TEL-AVIV 67899
Tel: 3 88 388
Telex: 33569 Motil IL
Cable: BASTEL Tel-Aviv
CH,CM,CS,E,M,P

ITALY

Hewlett-Packard Italiana S.p.A
Traversa 99C
Via Giulio Petroni, 19
I-70124 **BARI**
Tel: (080) 41-07-44
M

Hewlett-Packard Italiana S.p.A.
Via Martin Luther King, 38/III
I-40132 **BOLOGNA**
Tel: (051) 402394
Telex: 511630
CH,E,MS

Hewlett-Packard Italiana S.p.A.
Via Principe Nicola 43G/C
I-95126 **CATANIA**
Tel: (095) 37-10-87
Telex: 970291
C,P

Hewlett-Packard Italiana S.p.A.
Via G. Di Vittorio 9
I-20063 **CERNUSCO SUL NAVIGLIO**
(Milano)
Tel: (02) 923691
Telex: 334632
A,CH,CM,CS,E,MP,P

Hewlett-Packard Italiana S.p.A.
Via C. Colombo 49
I-20090 **TREZZANO SUL NAVIGLIO**
(Milano)
Tel: (02) 4459041
Telex: 322116
C,M

Hewlett-Packard Italiana S.p.A.
Via Nuova San Rocco a
Capodimonte, 62/A
I-80131 **NAPOLI**
Tel: (081) 7413544
Telex: 710698
A,CH,E

Hewlett-Packard Italiana S.p.A.
Viale G. Modugno 33
I-16156 **GENOVA PEGLI**
Tel: (010) 68-37-07
Telex: 215238
E,C

Hewlett-Packard Italiana S.p.A.
Via Pelizzo 15
I-35128 **PADOVA**
Tel: (049) 664888
Telex: 430315
A,CH,E,MS

Hewlett-Packard Italiana S.p.A.
Viale C. Pavese 340
I-00144 **ROMA EUR**
Tel: (06) 54831
Telex: 610514
A,CH,CM,CS,E,MS,P*

Hewlett-Packard Italiana S.p.A.
Via di Casellina 57/C
I-50018 **SCANDICCI-FIRENZE**
Tel: (055) 753863

Hewlett-Packard Italiana S.p.A.
Corso Svizzera, 185
I-10144 **TORINO**
Tel: (011) 74 4044
Telex: 221079
CH,E

JAPAN

Yokogawa-Hewlett-Packard Ltd.
152-1, Onna
ATSUGI, Kanagawa, 243
Tel: (0462) 28-0451
CM,C*,E

Yokogawa-Helwett-Packard Ltd.
Meiji-Seimei Bldg. 6F
3-1 Hon Chiba-Cho
CHIBA, 280
Tel: 472 25 7701
E,CH,CS

Yokogawa-Hewlett-Packard Ltd.
Yasuda-Seimei Hiroshima Bldg.
6-11, Hon-dori, Naka-ku
HIROSHIMA, 730
Tel: 82-241-0611

Yokogawa-Hewlett-Packard Ltd.
Towa Building
2-3, Kaigan-dori, 2 Chome Chuo-ku
KOBE, 650
Tel: (078) 392-4791
C,E

Yokogawa-Hewlett-Packard Ltd.
Kumagaya Asahi 82 Bldg
3-4 Tsukuba
KUMAGAYA, Saitama 360
Tel: (0485) 24-6563
CH,CM,E

Yokogawa-Hewlett-Packard Ltd.
Asahi Shinbun Daiichi Seimei Bldg.
4-7, Hanabata-cho
KUMAMOTO,860
Tel: (0963) 54-7311
CH,E

Yokogawa-Hewlett-Packard Ltd.
Shin-Kyoto Center Bldg.
614, Higashi-Shiokoji-cho
Karasuma-Nishiiru
Shiokoji-dori, Shimogyo-ku
KYOTO, 600
Tel: 075-343-0921
CH,E

Yokogawa-Hewlett-Packard Ltd.
Mito Mitsui Bldg
4-73, Sanno-maru, 1 Chome
MITO, Ibaraki 310
Tel: (0292) 25-7470
CH,CM,E

Yokogawa-Hewlett-Packard Ltd.
Sumitomo Seimei 14-9 Bldg.
Meieki-Minami, 2 Chome
Nakamura-ku
NAGOYA, 450
Tel: (052) 571-5171
CH,CM,CS,E,MS

Yokogawa-Hewlett-Packard Ltd.
Chuo Bldg.,
4-20 Nishinakajima, 5 Chome
Yodogawa-ku
OSAKA, 532
Tel: (06) 304-6021
Telex: YHPOSA 523-3624
A,CH,CM,CS,E,MP,P*

Yokogawa-Hewlett-Packard Ltd.
27-15, Yabe, 1 Chome
SAGAMIHARA Kanagawa, 229
Tel: 0427 59-1311

Yokogawa-Hewlett-Packard Ltd.
Daiichi Seimei Bldg.
7-1, Nishi Shinjuku, 2 Chome
Shinjuku-ku,**TOKYO** 160
Tel: 03-348-4611
CH,E

Yokogawa-Hewlett-Packard Ltd.
29-21 Takaido-Higashi, 3 Chome
Suginami-ku **TOKYO** 168
Tel: (03) 331-611
Telex: 232-2024 YHPTOK
A,CH,CM,CS,E,MP,P*

Yokogawa-Hewlett-Packard Ltd.
Daiichi Asano Building
2-8, Odori, 5 Chome
UTSUNOMIYA, Tochigi 320
Tel: (0286) 25-7155
CH,CS,E

Yokogawa-Hewlett-Packard Ltd.
Yasuda Seimei Nishiguchi Bldg.
30-4 Tsuruya-cho, 3 Chome
YOKOHAMA 221
Tel: (045) 312-1252
CH,CM,E

JORDAN

Mouasher Cousins Company
P.O. Box 1387
AMMAN
Tel: 24907, 39907
Telex: 21456 SABCO JO
CH,E,M,P

KENYA

AOCOM Ltd., Inc., Kenya
P.O.Box 30070
NAIROBI
Tel: 331955
Telex: 22639
E,M

KOREA

Samsung Electronics HP Division
12 Fl. Kinam Bldg.
San 75-31, Yeoksam-Dong
Kangnam-Ku
Yeongdong P.O. Box 72
SEOUL
Tel: 555-7555, 555-5447
Telex: K27364 SAMSAN
A,CH,CM,CS,E,M,P

KUWAIT

Al-Khaldiya Trading & Contracting
P.O. Box 830 Safat
KUWAIT
Tel: 42-4910, 41-1726
Telex: 22481 Areeg kt
CH,E,M

Photo & Cine Equipment
P.O. Box 270 Safat
KUWAIT
Tel: 42-2846, 42-3801
Telex: 22247 Matin kt
P

LEBANON

G.M. Dolmadjian
Achrafieh
P.O. Box 165.167
BEIRUT
Tel: 290293
*MP**
Computer Information Systems
P.O. Box 11-6274
BEIRUT
Tel: 89 40 73
Telex: 22259
C

LUXEMBOURG

Hewlett-Packard Belgium S.A./N.V.
Blvd de la Woluwe, 100
Woluwedal
B-1200 **BRUSSELS**
Tel: (02) 762-32-00
Telex: 23-494 paloben bru
A,CH,CM,CS,E,MP,P

MALAYSIA

Hewlett-Packard Sales (Malaysia)
Sdn. Bhd.
1st Floor, Bangunan British
American
Jalan Semantan, Damansara Heights
KUALA LUMPUR 23-03
Tel: 943022
Telex: MA31011
A,CH,E,M,P*

SALES & SUPPORT OFFICES

Arranged alphabetically by country

MAYLAYSIA (Cont'd)

Protel Engineering
P.O.Box 1917
Lot 6624, Section 64
23/4 Pending Road
*Kuching, **SARAWAK***
Tel: 36299
Telex: MA 70904 PROMAL
Cable: PROTELENG
A,E,M

MALTA

Philip Toledo Ltd.
Notabile Rd.
MRIEHEL
Tel: 447 47, 455 66
Telex: Media MW 649
E,P

MEXICO

Hewlett-Packard Mexicana, S.A. de C.V.
Av. Periferico Sur No. 6501
Tepepan, Xochimilco
16020 **MEXICO D.F.**
Tel: 6-76-46-00
Telex: 17-74-507 HEWPACK MEX
A,CH,CS,E,MS,P

Hewlett-Packard Mexicana, S.A. de C.V.
Ave. Colonia del Valle 409
Col. del Valle
Municipio de Garza Garcia
MONTERREY, Nuevo Leon
Tel: 78 42 41
Telex: 038 410
CH

ECISA
José Vasconcelos No. 218
Col. Condesa Deleg. Cuauhtémoc
***MEXICO** D.F. 06140*
Tel: 553-1206
Telex: 17-72755 ECE ME
M

MOROCCO

Dolbeau
81 rue Karatchi
CASABLANCA
Tel: 3041-82, 3068-38
Telex: 23051, 22822
E

Gerep
2 rue d'Agadir
Boite Postale 156
CASABLANCA
Tel: 272093, 272095
Telex: 23 739
P

NETHERLANDS

Hewlett-Packard Nederland B.V.
Van Heuven Goedhartlaan 121
NL 1181KK **AMSTELVEEN**
P.O. Box 667
NL1180 AR **AMSTELVEEN**
Tel: (020) 47-20-21
Telex: 13 216 HEPA NL
A,CH,CM,CS,E,MP,P

Hewlett-Packard Nederland B.V.
Bongerd 2
NL 2906VK **CAPELLE A/D IJSSEL**
P.O. Box 41
NL 2900AA **CAPELLE A/D IJSSEL**
Tel: (10) 51-64-44
Telex: 21261 HEPAC NL
A,CH,CS,E

Hewlett-Packard Nederland B.V.
Pastoor Petersstraat 134-136
NL 5612 LV **EINDHOVEN**
P.O. Box 2342
NL 5600 CH **EINDHOVEN**
Tel: (040) 326911
Telex: 51484 hepae nl
A,CH**,**,E,M

NEW ZEALAND

Hewlett-Packard (N.Z.) Ltd.
5 Owens Road
P.O. Box 26-189
Epsom, **AUCKLAND**
Tel: 687-159
Cable: HEWPACK Auckland
CH,CM,E,P*

Hewlett-Packard (N.Z.) Ltd.
4-12 Cruickshank Street
Kilbirnie, **WELLINGTON 3**
P.O. Box 9443
Courtenay Place, **WELLINGTON 3**
Tel: 877-199
Cable: HEWPACK Wellington
CH,CM,E,P

Northrop Instruments & Systems Ltd.
369 Khyber Pass Road
P.O. Box 8602
AUCKLAND
Tel: 794-091
Telex: 60605
A,M

Northrop Instruments & Systems Ltd.
110 Mandeville St.
P.O. Box 8388
CHRISTCHURCH
Tel: 486-928
Telex: 4203
A,M

Northrop Instruments & Systems Ltd.
Sturdee House
85-87 Ghuznee Street
P.O. Box 2406
WELLINGTON
Tel: 850-091
Telex: NZ 3380
A,M

NORTHERN IRELAND

See United Kingdom

NORWAY

Hewlett-Packard Norge A/S
Folke Bernadottes vei 50
P.O. Box 3558
N-5033 **FYLLINGSDALEN** (Bergen)
Tel: 0047/5/16 55 40
Telex: 16621 hpnas n
CH,CS,E,MS

Hewlett-Packard Norge A/S
Österndalen 16-18
P.O. Box 34
N-1345 **ÖSTERÅS**
Tel: 0047/2/17 11 80
Telex: 16621 hpnas n
A,CH,CM,CS,E,M,P

OMAN

Khimjil Ramdas
P.O. Box 19
MUSCAT
Tel: 722225, 745601
Telex: 3289 BROKER MB MUSCAT
P

Suhail & Saud Bahwan
P.O.Box 169
MUSCAT
Tel: 734 201-3
Telex: 3274 BAHWAN MB

PAKISTAN

Mushko & Company Ltd.
1-B, Street 43
Sector F-8/1
ISLAMABAD
Tel: 51071
Cable: FEMUS Rawalpindi
A,E,M

Mushko & Company Ltd.
Oosman Chambers
Abdullah Haroon Road
KARACHI 0302
Tel: 524131, 524132
Telex: 2894 MUSKO PK
Cable: COOPERATOR Karachi
A,E,M,P

PANAMA

Electrónico Balboa, S.A.
Calle Samuel Lewis, Ed. Alfa
Apartado 4929
PANAMA 5
Tel: 63-6613, 63-6748
Telex: 3483 ELECTRON PG
A,CM,E,M,P

PERU

Cía Electro Médica S.A.
Los Flamencos 145, San Isidro
Casilla 1030
LIMA 1
Tel: 41-4325, 41-3703
Telex: Pub. Booth 25306
CM,E,M,P

PHILIPPINES

The Online Advanced Systems Corporation
Rico House, Amorsolo Cor. Herrera Street
Legaspi Village, Makati
P.O. Box 1510
*Metro **MANILA***
Tel: 85-35-81, 85-34-91, 85-32-21
Telex: 3274 ONLINE
A,CH,CS,E,M

Electronic Specialists and Proponents Inc.
690-B Epifanio de los Santos Avenue
*Cubao, **QUEZON CITY***
P.O. Box 2649 Manila
Tel: 98-96-81, 98-96-82, 98-96-83
Telex: 40018, 42000 ITT GLOBE MACKAY BOOTH
P

PORTUGAL

Mundinter
Intercambio Mundial de Comércio S.A.R.L.
P.O. Box 2761
Av. Antonio Augusto de Aguiar 138
P-LISBON
Tel: (19) 53-21-31, 53-21-37
Telex: 16691 munter p
M

Soquimica
Av. da Liberdade, 220-2
*1298 **LISBOA** Codex*
Tel: 56 21 81/2/3
Telex: 13316 SABASA
P

Telectra-Empresa Técnica de Equipmentos Eléctricos S.A.R.L.
Rua Rodrigo da Fonseca 103
P.O. Box 2531
P-LISBON 1
Tel: (19) 68-60-72
Telex: 12598
CH,CS,E,P

PUERTO RICO

Hewlett-Packard Puerto Rico
Ave. Muñoz Rivera #101
Esq. Calle Ochoa
HATO REY, Puerto Rico 00918
Tel: (809) 754-7800

Hewlett-Packard Puerto Rico
Calle 272 Edificio 203
Urb. Country Club
RIO PIEDRAS, Puerto Rico
P.O. Box 4407
CAROLINA, Puerto Rico 00628
Tel: (809) 762-7255
A,CH,CS

QATAR

Computearbia
P.O. Box 2750
DOHA
Tel: 883555
Telex: 4806 CHPARB
P

Eastern Technical Services
P.O.Box 4747
DOHA
Tel: 329 993
Telex: 4156 EASTEC DH

Nasser Trading & Contracting
P.O.Box 1563
DOHA
Tel: 22170, 23539
Telex: 4439 NASSER DH
M

SAUDI ARABIA

Modern Electronic Establishment
Hewlett-Packard Division
P.O. Box 22015
Thuobah
AL-KHOBAR
Tel: 895-1760, 895-1764
Telex: 671 106 HPMEEK SJ
Cable: ELECTA AL-KHOBAR
CH,CS,E,M

Modern Electronic Establishment
Hewlett-Packard Division
P.O. Box 1228
Redec Plaza, 6th Floor
JEDDAH
Tel: 644 38 48
Telex: 4027 12 FARNAS SJ
Cable: ELECTA JEDDAH
CH,CS,E,M

Modern Electronic Establishment
Hewlett-Packard Division
P.O.Box 22015
RIYADH
Tel: 491-97 15, 491-63 87
Telex: 202049 MEERYD SJ
CH,CS,E,M

Abdul Ghani El Ajou
P.O. Box 78
RIYADH
Tel: 40 41 717
Telex: 200 932 EL AJOU
P

SCOTLAND

See United Kingdom

SINGAPORE

Hewlett-Packard Singapore (Sales) Pte. Ltd.
#08-00 Inchcape House
450-2 Alexandra Road
P.O. Box 58 Alexandra Rd. Post Office
SINGAPORE, 9115
Tel: 631788
Telex: HPSGSO RS 34209
Cable: HEWPACK, Singapore
A,CH,CS,E,MS,P

SALES & SUPPORT OFFICES

Arranged alphabetically by country

SINGAPORE (Cont'd)

Dynamar International Ltd.
Unit 05-11 Block 6
Kolam Ayer Industrial Estate
SINGAPORE 1334
Tel: 747-6188
Telex: RS 26283
CM

SOUTH AFRICA

Hewlett-Packard So Africa (Pty.) Ltd.
P.O. Box 120
Howard Place **CAPE PROVINCE** 7450
Pine Park Center, Forest Drive,
Pinelands
CAPE PROVINCE 7405
Tel: 53-7954
Telex: 57-20006
A,CH,CM,E,MS,P

Hewlett-Packard So Africa (Pty.) Ltd.
P.O. Box 37099
92 Overport Drive
DURBAN 4067
Tel: 28-4178, 28-4179, 28-4110
Telex: 6-22954
CH,CM

Hewlett-Packard So Africa (Pty.) Ltd.
6 Linton Arcade
511 Cape Road
Linton Grange
PORT ELIZABETH 6000
Tel: 041-302148
CH

Hewlett-Packard So Africa (Pty.) Ltd.
P.O.Box 33345
Glenstantia 0010 **TRANSVAAL**
1st Floor East
Constantia Park Ridge Shopping
Centre
Constantia Park
PRETORIA
Tel: 982043
Telex: 32163
CH,E

Hewlett-Packard So Africa (Pty.) Ltd.
Private Bag Wendywood
SANDTON 2144
Tel: 802-5111, 802-5125
Telex: 4-20877
Cable: HEWPACK Johannesburg
A,CH,CM,CS,E,MS,P

SPAIN

Hewlett-Packard Española S.A.
Calle Entenza, 321
E-BARCELONA 29
Tel: 322.24.51, 321.73.54
Telex: 52603 hpbee
A,CH,CS,E,MS,P

Hewlett-Packard Española S.A.
Calle San Vicente S/No
Edificio Albia II
E-BILBAO 1
Tel: 423.83.06
A,CH,E,MS

Hewlett-Packard Española S.A.
Crta. de la Coruña, Km. 16, 400
Las Rozas
E-MADRID
Tel: (1) 637.00.11
CH,CS,M

Hewlett-Packard Española S.A.
Avda. S. Francisco Javier, S/no
Planta 10. Edificio Sevilla 2,
E-SEVILLA 5
Tel: 64.44.54
Telex: 72933
A,CS,MS,P

Hewlett-Packard Española S.A.
Calle Ramon Gordillo, 1 (Entlo.3)
E-VALENCIA 10
Tel: 361-1354
CH,P

SWEDEN

Hewlett-Packard Sverige AB
Sunnanvagen 14K
S-22226 **LUND**
Tel: (046) 13-69-79
Telex: (854) 17886 (via Spånga
office)
CH

Hewlett-Packard Sverige AB
Östra Tullgatan 3
S-21128 **MALMÖ**
Tel: (040) 70270
Telex: (854) 17886 (via Spånga
office)

Hewlett-Packard Sverige AB
Västra Vintergatan 9
S-70344 **ÖREBRO**
Tel: (19) 10-48-80
Telex: (854) 17886 (via Spånga
office)
CH

Hewlett-Packard Sverige AB
Skalholtsgatan 9, Kista
Box 19
S-16393 **SPÅNGA**
Tel: (08) 750-2000
Telex: (854) 17886
Telefax: (08) 7527781
A,CH,CM,CS,E,MS,P

Hewlett-Packard Sverige AB
Frötallisgatan 30
S-42132 **VÄSTRA-FRÖLUNDA**
Tel: (031) 49-09-50
Telex: (854) 17886 (via Spånga
office)
CH,E,P

SWITZERLAND

Hewlett-Packard (Schweiz) AG
Clarastrasse 12
CH-4058 **BASEL**
Tel: (61) 33-59-20
A

Hewlett-Packard (Schweiz) AG
7, rue du Bois-du-Lan
Case Postale 365
CH-1217 **MEYRIN** 2
Tel: (0041) 22-83-11-11
Telex:27333 HPAG CH
CH,CM,CS

Hewlett-Packard (Schweiz) AG
Allmend 2
CH-8967 **WIDEN**
Tel: (0041) 57 31 21 11
Telex: 53933 hpag ch
Cable: HPAG CH
A,CH,CM,CS,E,MS,P

SYRIA

General Electronic Inc.
Nuri Basha Ahnaf Ebn Kays Street
P.O. Box 5781
DAMASCUS
Tel: 33-24-87
Telex: 411 215
Cable: ELECTROBOR DAMASCUS
E

Middle East Electronics
P.O.Box 2308
Abu Rumnaneh
DAMASCUS
Tel: 33 4 5 92
Telex: 411 304
M

TAIWAN

Hewlett-Packard Far East Ltd.
Kaohsiung Office
2/F 68-2, Chung Cheng 3rd Road
KAOHSIUNG
Tel: (07) 241-2318
CH,CS,E

Hewlett-Packard Far East Ltd.
Taiwan Branch
8th Floor
337 Fu Hsing North Road
TAIPEI
Tel: (02) 712-0404
Telex: 24439 HEWPACK
Cable:HEWPACK Taipei
A,CH,CM,CS,E,M,P

Ing Lih Trading Co.
3rd Floor, 7 Jen-Ai Road, Sec. 2
TAIPEI 100
Tel: (02) 3948191
Cable: INGLIH TAIPEI
A

THAILAND

Unimesa
30 Patpong Ave., Suriwong
BANGKOK 5
Tel: 235-5727
Telex: 84439 Simonco TH
Cable: UNIMESA Bangkok
A,CH,CS,E,M

Bangkok Business Equipment Ltd.
5/5-6 Dejo Road
BANGKOK
Tel: 234-8670, 234-8671
Telex: 87669-BEQUIPT TH
Cable: BUSIQUIPT Bangkok
P

TRINIDAD & TOBAGO

Caribbean Telecoms Ltd.
50/A Jerningham Avenue
P.O. Box 732
PORT-OF-SPAIN
Tel: 62-44213, 62-44214
Telex: 235,272 HUGCO WG
CM,E,M,P

TUNISIA

Tunisie Electronique
31 Avenue de la Liberte
TUNIS
Tel: 280-144
E,P

Corema
1 ler. Av. de Carthage
TUNIS
Tel: 253-821
Telex: 12319 CABAM TN
M

TURKEY

Teknim Company Ltd.
Iran Caddesi No. 7
*Kavaklidere, **ANKARA***
Tel: 275800
Telex: 42155 TKNM TR
E

E.M.A.
Medina Eldem Sokak No.41/6
Yuksel Caddesi
ANKARA
Tel: 175 622
Telex: 42 591
M

UNITED ARAB EMIRATES

Emilac Ltd.
P.O. Box 2711
ABU DHABI
Tel: 82 04 19-20
Cable: EMITAC ABUDHABI

Emilac Ltd.
P.O. Box 1641
SHARJAH
Tel: 591 181
Telex: 68136 Emilac Sh
CH,CS,E,M,P

UNITED KINGDOM

GREAT BRITAIN

Hewlett-Packard Ltd.
Trafalgar House
Navigation Road
ALTRINCHAM
Cheshire WA14 1NU
Tel: 061 928 6422
Telex: 668068
A,CH,CS,E,M,MS,P

Hewlett-Packard Ltd.
Elstree House, Elstree Way
BOREHAMWOOD, Herts WD6 1SG
Tel: 01 207 5000
Telex: 8952716
E,CH,CS,P

Hewlett-Packard Ltd.
Oakfield House, Oakfield Grove
Clifton **BRISTOL,** Avon BS8 2BN
Tel: 0272 736806
Telex: 444302
CH,CS,E,P

Hewlett-Packard Ltd.
Bridewell House
Bridewell Place
LONDON EC4V 6BS
Tel: 01 583 6565
Telex: 298163
CH,CS,P

Hewlett-Packard Ltd.
Fourier House
257-263 High Street
LONDON COLNEY
Herts. AL2 1HA, St. Albans
Tel: 0727 24400
Telex: 1-8952716
CH,CS

Hewlett-Packard Ltd.
Pontefract Road
NORMANTON, West Yorkshire WF6 1RN
Tel: 0924 895566
Telex: 557355
CH,CS,P

Hewlett-Packard Ltd.
The Quadrangle
106-118 Station Road
REDHILL, Surrey RH1 1PS
Tel: 0737 68655
Telex: 947234
CH,CS,E,P

SALES & SUPPORT OFFICES

Arranged alphabetically by country

GREAT BRITAIN (Cont'd)
Hewlett-Packard Ltd.
Avon House
435 Stratford Road
Shirley, **SOLIHULL**, West Midlands
890 48L
Tel: 021 745 8800
Telex: 339105
CH,CS,E,P

Hewlett-Packard Ltd.
West End House
41 High Street, West End
SOUTHAMPTON
Hampshire S03 3DQ
Tel: 04218 6767
Telex: 477138
CH,CS,P

Hewlett-Packard Ltd.
Eskdale Rd.
Winnersh, **WOKINGHAM**
Berkshire RG11 5DZ
Tel: 0734 696622
Telex: 848884
E

Hewlett-Packard Ltd.
King Street Lane
Winnersh, **WOKINGHAM**
Berkshire RG11 5AR
Tel: 0734 784774
Telex: 847178
A,CH,CS,E,M,MP,P

Hewlett-Packard Ltd.
Nine Mile Ride
Easthampstead, **WOKINGHAM**
Berkshire, 3RG11 3LL
Tel: 0344 773100
Telex: 848805
CH,CS,E,P

IRELAND

NORTHERN IRELAND
Hewlett-Packard Ltd.
Cardiac Services Building
95A Finaghy Road South
BELFAST BT10 0BY
Tel: 0232 625-566
Telex: 747626
CH,CS

SCOTLAND
Hewlett-Packard Ltd.
SOUTH QUEENSFERRY
West Lothian, EH30 9TG
Tel: 031 331 1188
Telex: 72682
CH,CM,CS,E,M,P

UNITED STATES

Alabama
Hewlett-Packard Co.
700 Century Park South, Suite 128
BIRMINGHAM, AL 35226
Tel: (205) 822-6802
A,CH,M

Hewlett-Packard Co.
420 Wynn Drive
HUNTSVILLE, AL 35805
P.O. Box 7700
HUNTSVILLE, AL 35807
Tel: (205) 830-2000
CH,CM,CS,E,M*

Arizona
Hewlett-Packard Co.
8080 Pointe Parkway West
PHOENIX, AZ 85044
Tel: (602) 273-8000
A,CH,CM,CS,E,MS

Hewlett-Packard Co.
2424 East Aragon Road
TUCSON, AZ 85706
Tel: (602) 889-4631
CH,E,MS**

California
Hewlett-Packard Co.
99 South Hill Dr.
BRISBANE, CA 94005
Tel: (415) 330-2500
CH,CS

Hewlett-Packard Co.
P.O. Box 7830 (93747)
5060 E. Clinton Avenue, Suite 102
FRESNO, CA 93727
Tel: (209) 252-9652
CH,CS,MS

Hewlett-Packard Co.
P.O. Box 4230
1430 East Orangethorpe
FULLERTON, CA 92631
Tel: (714) 870-1000
CH,CM,CS,E,MP

Hewlett-Packard Co.
320 S. Kellogg, Suite B
GOLETA, CA 93117
Tel: (805) 967-3405
CH

Hewlett-Packard Co.
5400 W. Rosecrans Boulevard
LAWNDALE, CA 90260
P.O. Box 92105
LOS ANGELES, CA 90009
Tel: (213) 970-7500
Telex: 910-325-6608
CH,CM,CS,MP

Hewlett-Packard Co.
3155 Porter Oaks Drive
PALO ALTO, CA 94304
Tel: (415) 857-8000
CH,CS,E

Hewlett-Packard Co.
4244 So. Market Court, Suite A
P.O. Box 15976
SACRAMENTO, CA 95852
Tel: (916) 929-7222
A*,CH,CS,E,MS

Hewlett-Packard Co.
9606 Aero Drive
P.O. Box 23333
SAN DIEGO, CA 92139
Tel: (619) 279-3200
CH,CM,CS,E,MP

Hewlett-Packard Co.
2305 Camino Ramon "C"
SAN RAMON, CA 94583
Tel: (415) 838-5900
CH,CS

Hewlett-Packard Co.
3005 Scott Boulevard
SANTA CLARA, CA 95050
Tel: (408) 988-7000
Telex: 910-338-0586
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
5703 Corsa Avenue
WESTLAKE VILLAGE, CA 91362
Tel: (213) 706-6800
E*,CH*,CS*

Colorado
Hewlett-Packard Co.
24 Inverness Place, East
ENGLEWOOD, CO 80112
Tel: (303) 649-5000
A,CH,CM,CS,E,MS

Connecticut
Hewlett-Packard Co.
47 Barnes Industrial Road South
P.O. Box 5007
WALLINGFORD, CT 06492
Tel: (203) 265-7801
A,CH,CM,CS,E,MS

Florida
Hewlett-Packard Co.
2901 N.W. 62nd Street
P.O. Box 24210
FORT LAUDERDALE, FL 33307
Tel: (305) 973-2600
CH,CS,E,MP

Hewlett-Packard Co.
6177 Lake Ellenor Drive
P.O. Box 13910
ORLANDO, FL 32859
Tel: (305) 859-2900
A,CH,CM,CS,E,MS

Hewlett-Packard Co.
5750B N. Hoover Blvd., Suite 123
P.O. Box 15200
TAMPA, FL 33614
Tel: (813) 884-3282
A*,CH,CM,CS,E*,M*

Georgia
Hewlett-Packard Co.
2000 South Park Place
P.O. Box 105005
ATLANTA, GA 30348
Tel: (404) 955-1500
Telex: 810-766-4890
A,CH,CM,CS,E,MP

Hawaii
Hewlett-Packard Co.
Kawaiahao Plaza, Suite 190
567 South King Street
HONOLULU, HI 96813
Tel: (808) 526-1555
A,CH,E,MS

Illinois
Hewlett-Packard Co.
304 Eldorado Road
P.O. Box 1607
BLOOMINGTON, IL 61701
Tel: (309) 662-9411
CH,MS**

Hewlett-Packard Co.
1100 31st Street, Suite 100
DOWNERS GROVE, IL 60515
Tel: (312) 960-5760
CH,CS

Hewlett-Packard Co.
5201 Tollview Drive
ROLLING MEADOWS, IL 60008
Tel: (312) 255-9800
Telex: 910-687-1066
A,CH,CM,CS,E,MP

Indiana
Hewlett-Packard Co.
7301 No. Shadeland Avenue
P.O. Box 50807
INDIANAPOLIS, IN 46250
Tel: (317) 842-1000
A,CH,CM,CS,E,MS

Iowa
Hewlett-Packard Co.
1776 22nd Street, Suite 1
WEST DES MOINES, IA 50265
Tel: (515) 224-1435
CH,MS**

Kansas
Hewlett-Packard Co.
7804 East Funston Road, #203
WICHITA, KS 67207
Tel: (316) 684-8491
CH

Kentucky
Hewlett-Packard Co.
10300 Linn Station Road, #100
LOUISVILLE, KY 40223
Tel: (502) 426-0100
A,CH,CS,MS

Louisiana
Hewlett-Packard Co.
160 James Drive East
ST. ROSE, LA 70087
P.O. Box 1449
KENNER, LA 70063
Tel: (504) 467-4100
A,CH,CS,E,MS

Maryland
Hewlett-Packard Co.
3701 Koppers Street
BALTIMORE, MD 21227
Tel: (301) 644-5800
Telex: 710-862-1943
A,CH,CM,CS,E,MS

Hewlett-Packard Co.
2 Choke Cherry Road
ROCKVILLE, MD 20850
Tel: (301) 948-6370
A,CH,CM,CS,E,MP

Massachusetts
Hewlett-Packard Co.
1775 Minuteman Road
ANDOVER, MA 01810
Tel: (617) 682-1500
A,C,CH,CS,CM,E,MP,P*

Hewlett-Packard Co.
32 Hartwell Avenue
LEXINGTON, MA 02173
Tel: (617) 861-8960
CH,CS,E

Michigan
Hewlett-Packard Co.
4326 Cascade Road S.E.
GRAND RAPIDS, MI 49506
Tel: (616) 957-1970
CH,CS,MS

Hewlett-Packard Co.
1771 W. Big Beaver Road
TROY, MI 48084
Tel: (313) 643-6474
CH,CS

Minnesota
Hewlett-Packard Co.
2025 W. Larpenteur Ave.
ST. PAUL, MN 55113
Tel: (612) 644-1100
A,CH,CM,CS,E,MP

Missouri
Hewlett-Packard Co.
11131 Colorado Avenue
KANSAS CITY, MO 64137
Tel: (816) 763-8000
A,CH,CM,CS,E,MS

Hewlett-Packard Co.
13001 Hollenberg Drive
BRIDGETON, MO 63044
Tel: (314) 344-5100
A,CH,CS,E,MP

SALES & SUPPORT OFFICES

Arranged alphabetically by country

UNITED STATES (Cont'd)

Nebraska
Hewlett-Packard
10824 Old Mill Rd., Suite 3
OMAHA, NE 68154
Tel: (402) 334-1813
CM,MS

New Jersey
Hewlett-Packard Co.
120 W. Century Road
PARAMUS, NJ 07652
Tel: (201) 265-5000
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
60 New England Av. West
PISCATAWAY, NJ 08854
Tel: (201) 981-1199
A,CH,CM,CS,E

New Mexico
Hewlett-Packard Co.
11300 Lomas Blvd.,N.E.
P.O. Box 11634
ALBUQUERQUE, NM 87112
Tel: (505) 292-1330
CH,CS,E,MS

New York
Hewlett-Packard Co.
5 Computer Drive South
ALBANY, NY 12205
Tel: (518) 458-1550
A,CH,E,MS

Hewlett-Packard Co.
9600 Main Street
P.O. Box AC
CLARENCE, NY 14031
Tel: (716) 759-8621
CH

Hewlett-Packard Co.
200 Cross Keys Office Park
FAIRPORT, NY 14450
Tel: (716) 223-9950
CH,CM,CS,E,MS

Hewlett-Packard Co.
7641 Henry Clay Blvd.
LIVERPOOL, NY 13088
Tel: (315) 451-1820
A,CH,CM,E,MS

Hewlett-Packard Co.
No. 1 Pennsylvania Plaza
55th Floor
34th Street & 8th Avenue
MANHATTAN NY 10119
Tel: (212) 971-0800
CH,CS,E*,M*

Hewlett-Packard Co.
250 Westchester Avenue
WHITE PLAINS, NY 10604
Tel: (914) 684-6100
CM,CH,CS,E

Hewlett-Packard Co.
3 Crossways Park West
WOODBURY, NY 11797
Tel: (516) 921-0300
A,CH,CM,CS,E,MS

North Caroiina
Hewlett-Packard Co.
5605 Roanne Way
P.O. Box 26500
GREENSBORO, NC 27420
Tel: (919) 852-1800
A,CH,CM,CS,E,MS

Ohio
Hewlett-Packard Co.
9920 Carver Road
CINCINNATI, OH 45242
Tel: (513) 891-9870
CH,CS,MS

Hewlett-Packard Co.
16500 Sprague Road
CLEVELAND, OH 44130
Tel: (216) 243-7300
A,CH,CM,CS,E,MS

Hewlett-Packard Co.
962 Crupper Ave.
COLUMBUS, OH 43229
Tel: (614) 436-1041
Eff: Nov. 25, 1983
675 Brooksedge Blvd.
WESTERVILLE, OH 43081
CH,CM,CS,E*

Hewlett-Packard Co.
330 Progress Rd.
DAYTON, OH 45449
Tel: (513) 859-8202
A,CH,CM,E*,MS

Oklahoma
Hewlett-Packard Co.
304 N. Meridian, Suite A
P.O. Box 75609
OKLAHOMA CITY, OK 73147
Tel: (405) 946-9499
A*,CH,E*,MS

Hewlett-Packard Co.
3840 S. 103rd E. Avenue, #100
P.O. Box 35747
TULSA, OK 74153
Tel: (918) 665-3300
A**,CH,CS,M*

Oregon
Hewlett-Packard Co.
9255 S. W. Pioneer Court
P.O. Box 328
WILSONVILLE, OR 97070
Tel: (503) 682-8000
A,CH,CS,E*,MS

Pennsylvania
Hewlett-Packard Co.
111 Zeta Drive
PITTSBURGH, PA 15238
Tel: (412) 782-0400
A,CH,CS,E,MP

Hewlett-Packard Co.
2750 Monroe Boulevard
P.O. Box 713
VALLEY FORGE, PA 19482
Tel: (215) 666-9000
A,CH,CM,E,M

South Caroiina
Hewlett-Packard Co.
Brookside Park, Suite 122
1 Harbison Way
P.O. Box 21708
COLUMBIA, SC 29221
Tel: (803) 732-0400
CH,E,MS

Hewlett-Packard Co.
Koger Executive Center
Chesterfield Bldg., Suite 124
GREENVILLE, SC 29615
Tel: (803) 297-4120

Tennessee
Hewlett-Packard Co.
224 Peters Road, Suite 102
P.O. Box 22490
KNOXVILLE, TN 37922
Tel: (615) 691-2371
A*,CH,MS

Hewlett-Packard Co.
3070 Directors Row
MEMPHIS, TN 38131
Tel: (901) 346-8370
A,CH,MS

Texas
Hewlett-Packard Co.
4171 North Mesa
Suite C-110
EL PASO, TX 79902
Tel: (915) 533-3555
CH,E*,MS**

Hewlett-Packard Co.
10535 Harwin Drive
P.O. Box 42816
HOUSTON, TX 77042
Tel: (713) 776-6400
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
930 E. Campbell Rd.
P.O. Box 1270
RICHARDSON, TX 75080
Tel: (214) 231-6101
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
1020 Central Parkway South
P.O. Box 32993
SAN ANTONIO, TX 78216
Tel: (512) 494-9336
CH,CS,E,MS

Utah
Hewlett-Packard Co.
3530 W. 2100 South
SALT LAKE CITY, UT 84119
Tel: (801) 974-1700
A,CH,CS,E,MS

Virginia
Hewlett-Packard Co.
4305 Cox Road
GLEN ALLEN, VA 23060
P.O. Box 9669
RICHMOND, VA 23228
Tel: (804) 747-7750
A,CH,CS,E,MS

Washington
Hewlett-Packard Co.
15815 S.E. 37th Street
BELLEVUE, WA 98006
Tel: (206) 643-4000
A,CH,CM,CS,E,MP

Hewlett-Packard Co.
Suite A
708 North Argonne Road
SPOKANE, WA 99212
Tel: (509) 922-7000
CH,CS

West Virginia
Hewlett-Packard Co.
4604 MacCorkle Ave.
P.O. Box 4297
CHARLESTON, WV 25304
Tel: (304) 925-0492
A,MS

Wisconsin
Hewlett-Packard Co.
150 S. Sunny Slope Road
BROOKFIELD, WI 53005
Tel: (414) 784-8800
A,CH,CS,E*,MP

URUGUAY

Pablo Ferrando S.A.C. e I.
Avenida Italia 2877
Casilla de Correo 370
MONTEVIDEO
Tel: 80-2586
Telex: Public Booth 901
A,CM,E,M

VENEZUELA

Hewlett-Packard de Venezuela C.A.
3RA Transversal Los Ruices Norte
Edificio Segre 1, 2 & 3
Apartado 50933
CARACAS 1071
Tel: 239-4133
Telex: 251046 HEWPACK
A,CH,CS,E,MS,P

Hewlett-Packard de Venezuela C.A.
Calle-72-Entre 3H y 3Y, No. 3H-40
Edificio Ada-Evelyn, Local B
Apartado 2646
4001, **MARACAIBO**, Estado Zulia
Tel: (061) 80.304
C,E*

Hewlett-Packard de Venezuela C.A.
Calle Vargas Rondon
Edificio Seguros Carabobo, Piso 10
VALENCIA
Tel: (041) 51 385
CH,CS,P

Bioelectronica Medica C.A.
Calle Buen Pastor
Edif. Cota Mil-Piso 2 y Semi Sotano 1
Boleita Norte
Apartado 50710 ***CARACAS*** *1050A*
Tel: 239 84 41
Telex: 26518

ZIMBABWE

Field Technical Sales
45 Kelvin Road, North
P.B. 3458
SALISBURY
Tel: 705 231
Telex: 4-122 RH
C,E,M,P

July 1983 5952-6900

Indicates main office

HP distributors are printed in italics.

64620-90905, MARCH 1984
Replaces: 64620-90903, November 1983

PRINTED IN U.S.A.